日　本　国　特　許　庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出願年月日
Date of Application: ２００３年　３月１４日

出願番号
Application Number: 特願２００３－０６９３７５

[ST.10/C]: [ＪＰ２００３－０６９３７５]

出願人
Applicant(s): 株式会社産学連携機構九州
株式会社エイシーエス

PRIORITY DOCUMENT
SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)

２００３年　６月２０日

特許庁長官
Commissioner,
Japan Patent Office 太田信一郎

出証番号　出証特２００３－３０４８４０７

【書類名】　特許願

【整理番号】　H0301-01

【あて先】　特許庁長官　殿

【国際特許分類】　G06F 15/00

H04L 9/32

【発明者】

【住所又は居所】　福岡県福岡市東区箱松４－２２－１４　松原寮Ａ－３０２

【氏名】　今本　健二

【発明者】

【住所又は居所】　東京都町田市三輪緑山１－３－２　緑山ヒルズ２１２

【氏名】　大河　克好

【特許出願人】

【識別番号】　800000035

【氏名又は名称】　株式会社　産学連携機構九州

【特許出願人】

【識別番号】　500196087

【氏名又は名称】　株式会社　エイシーエス

【代理人】

【識別番号】　100096862

【弁理士】

【氏名又は名称】　清水　千春

【電話番号】　03-3543-0036

【選任した代理人】

【識別番号】　100067046

【弁理士】

【氏名又は名称】　尾股　行雄

【電話番号】　03-3543-0036

【手数料の表示】

【予納台帳番号】　057761

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　ワンタイムＩＤの生成方法、認証方法、認証システム、サーバ、クライアントおよびプログラム

【特許請求の範囲】

【請求項１】

複数の装置間またはアプリケーション間における認証において一回限り使用可能な識別情報をワンタイムＩＤとして、当該ワンタイムＩＤを生成する方法であって、

上記認証を行う装置またはアプリケーションの各々において、上記認証が必要な所定の通信単位毎に変化する可変共有鍵を生成するとともに、この可変共有鍵を引数とする一方向関数の関数値を求め、この関数値から上記ワンタイムＩＤを生成するようにしたことを特徴とするワンタイムＩＤの生成方法。

【請求項２】

複数の装置間またはアプリケーション間における認証において一回限り使用可能な識別情報をワンタイムＩＤとして、当該ワンタイムＩＤを生成する方法であって、

上記認証を行う装置またはアプリケーションの各々において、上記認証が必要な所定の通信単位毎に変化する可変共有鍵を生成するとともに、この可変共有鍵と通信順序または回数に関する情報とを引数とする一方向関数の関数値を求め、この関数値から上記ワンタイムＩＤを生成するようにしたことを特徴とするワンタイムＩＤの生成方法。

【請求項３】

複数の装置間またはアプリケーション間における認証において一回限り使用可能な識別情報をワンタイムＩＤとして、当該ワンタイムＩＤを生成する方法であって、

上記認証を行う装置またはアプリケーションの各々において、上記認証が必要な所定の通信単位内で乱数を生成するとともに、この乱数と所定の共有鍵とを引数とする一方向関数の関数値を求め、この関数値から上記ワンタイムＩＤを生成するようにしたことを特徴とするワンタイムＩＤの生成方法。

出証特２００３－３０４８４０７

【請求項４】

一方の装置と他方の装置間における認証において一回限り使用可能な識別情報をワンタイムＩＤとして、当該ワンタイムＩＤを双方の装置で生成するとともに、一方の装置が他方の装置にワンタイムＩＤを送信して、他方の装置が、一方の装置から受信したワンタイムＩＤと自らが生成したワンタイムＩＤとの比較・照合により、他方の装置を識別或いは認証する場合において、一方の装置および他方の装置がワンタイムＩＤを生成する方法であって、

一方の装置および他方の装置は、上記認証が必要な所定の通信単位毎に変化する可変共有鍵を生成するとともに、この可変共有鍵を引数とする一方向関数の関数値を求め、この関数値から上記ワンタイムＩＤを生成するようにしたことを特徴とするワンタイムＩＤの生成方法。

【請求項５】

一方の装置と他方の装置間における認証において一回限り使用可能な識別情報をワンタイムＩＤとして、当該ワンタイムＩＤを双方の装置で生成するとともに、一方の装置が他方の装置にワンタイムＩＤを送信して、他方の装置が、一方の装置から受信したワンタイムＩＤと自らが生成したワンタイムＩＤとの比較・照合により、他方の装置を識別或いは認証する場合において、一方の装置および他方の装置がワンタイムＩＤを生成する方法であって、

一方の装置および他方の装置は、上記認証が必要な所定の通信単位毎に変化する可変共有鍵を生成するとともに、この可変共有鍵と通信順序または回数に関する情報とを引数とする一方向関数の関数値を求め、この関数値から上記ワンタイムＩＤを生成するようにしたことを特徴とするワンタイムＩＤの生成方法。

【請求項６】

一方の装置と他方の装置間における認証において一回限り使用可能な識別情報をワンタイムＩＤとして、当該ワンタイムＩＤを双方の装置で生成するとともに、一方の装置が他方の装置にワンタイムＩＤを送信して、他方の装置が、一方の装置から受信したワンタイムＩＤと自らが生成したワンタイムＩＤとの比較・照合により、他方の装置を識別或いは認証する場合において、一方の装置および他方の装置がワンタイムＩＤを生成する方法であって、

一方の装置および他方の装置は、上記認証が必要な所定の通信単位内で乱数を生成するとともに、この乱数と所定の共有鍵とを引数とする一方向関数の関数値を求め、この関数値から上記ワンタイムＩＤを生成するようにしたことを特徴とするワンタイムＩＤの生成方法。

【請求項７】

請求項１または請求項４に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いて、第一装置と第二装置間における認証を行う認証方法であって、

上記第一装置が、上記第二装置との間で予め共有化された可変共有鍵を用いて上記ワンタイムＩＤを生成するとともに、この生成したワンタイムＩＤと、当該第一装置に予め設定されたＩＤを少なくとも引数とする一方向関数Ｆｃの関数値と、当該第一装置に予め記憶されたDiffie-Hellman公開値の一方とを上記第二装置に対して送信するステップと、

上記第二装置が、上記ワンタイムＩＤおよび上記一方向関数Ｆｃの関数値を演算により求め、この演算結果と、上記第一装置から受信したワンタイムＩＤおよび一方向関数Ｆｃの関数値との照合により、上記第一装置の正当性を判定するステップと、

上記第二装置が、上記第一装置を正当であると判定した場合に、当該第二装置に予め設定されたＩＤを少なくとも引数とする一方向関数Ｆｓの関数値と、当該第二装置に予め記憶されたDiffie-Hellman公開値の他方とを上記第一装置に対して送信するステップと、

上記第一装置が、上記一方向関数Ｆｓの関数値を演算により求め、この演算結果と、上記第二装置から受信した一方向関数Ｆｓの関数値との照合により、上記第二装置の正当性を判定するステップとを有することを特徴とする認証方法。

【請求項８】

上記一方向関数Ｆｃとして、所定の共有鍵、上記Diffie-Hellman公開値の一方、上記第一装置に予め設定されたＩＤ、上記ワンタイムＩＤを引数とする疑似乱数関数を用いるとともに、

上記一方向関数Ｆｓとして、上記所定の共有鍵、上記Diffie-Hellman公開値の

一方、上記Diffie-Hellman公開値の他方、上記第二装置に予め設定されたＩＤ、上記ワンタイムＩＤを引数とする疑似乱数関数を用いるようにしたことを特徴とする請求項７に記載の認証方法。

【請求項９】

請求項２または請求項５に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いて、第一装置と第二装置間における認証を行う認証方法であって、

上記第一装置が、上記第二装置との間で予め共有化された第一の可変共有鍵と当該第一装置の通信順序に関する情報とを引数とする一方向関数の関数値を第一のワンタイムＩＤとして生成するとともに、上記第一の可変共有鍵を用いて、当該第一装置に予め設定されたＩＤ、上記第二装置に予め設定されたＩＤ、当該第一装置に予め記憶されたDiffie-Hellman公開値の一方および上記第一のワンタイムＩＤを暗号化し、この暗号化データと上記第一のワンタイムＩＤとを上記第二装置に対して送信するステップと、

上記第二装置が、上記第一のワンタイムＩＤを演算により求め、この演算結果と、上記第一装置から受信した上記第一のワンタイムＩＤとの照合により、上記第一装置を識別するステップと、

上記第二装置が、上記第一装置を識別できた場合に、上記第一の可変共有鍵を用いて上記暗号化データを復号し、この復号したデータに含まれる、上記第一装置に予め設定されたＩＤ、当該第二装置に予め設定されたＩＤおよび上記第一のワンタイムＩＤに基づいて、上記第一装置の正当性を判定するステップと、

上記第二装置が、上記第一装置を正当であると判定した場合に、上記第一の可変共有鍵と当該第二装置の通信順序に関する情報とを引数とする一方向関数の関数値を第二のワンタイムＩＤとして生成するとともに、上記第一装置から受信したDiffie-Hellman公開値の一方と当該第二装置に予め記憶されたDiffie-Hellman公開値の他方とからDiffie-Hellman共通鍵を第二の可変共有鍵として生成し、この第二の可変共有鍵、上記第一装置に予め設定されたＩＤ、当該第二装置に予め設定されたＩＤおよび上記第二のワンタイムＩＤを引数とする一方向関数ｈの関数値と、上記Diffie-Hellman公開値の他方と、上記第二のワンタイムＩＤとを上

記第一装置に対して送信するステップと、

上記第一装置が、上記第二のワンタイムＩＤを演算により求め、この演算結果と、上記第二装置から受信した上記第二のワンタイムＩＤとの照合により、上記第二装置を識別するステップと、

上記第一装置が、上記第二装置を識別できた場合に、上記第二装置から受信した上記Diffie-Hellman公開値の他方と当該第一装置に予め記憶された上記Diffie-Hellman公開値の一方とからDiffie-Hellman共通鍵を上記第二の可変共有鍵として生成するとともに、この第二の可変共有鍵を用いて上記一方向関数ｈの関数値を演算により求め、この演算結果と、上記第二装置から受信した一方向関数ｈの関数値との照合により、上記第二装置の正当性を判定するステップとを有することを特徴とする認証方法。

【請求項１０】

上記第二のワンタイムＩＤを生成する一方向関数として、上記第一のワンタイムＩＤを生成する一方向関数とは異なる一方向関数を用いるようにしたことを特徴とする請求項９に記載の認証方法。

【請求項１１】

請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いて、第一装置と第二装置間における認証を行う認証方法であって、

上記第一装置が、第一の乱数を生成するとともに、上記第二装置との間で予め共有化された第一の共有鍵を引数とする一方向関数の関数値を第一のワンタイムＩＤとして求め、この第一のワンタイムＩＤと上記第一の乱数とを上記第二装置に対して送信するステップと、

上記第二装置が、第二の乱数を生成するとともに、上記第一の乱数と上記第一の共有鍵とを引数とする一方向関数の関数値を第二のワンタイムＩＤとして求め、この第二のワンタイムＩＤと上記第二の乱数とを上記第一装置に対して送信するステップと、

上記第一装置が、上記第一の乱数および上記第一の共有鍵に基づいて上記第二のワンタイムＩＤを演算により求め、この演算結果と上記第二装置から受信した

上記第二のワンタイムＩＤとの比較により、上記第二装置の正当性を判定するステップと、

上記第一装置が、上記第一の乱数および上記第二の乱数に基づいて第二の共有鍵を生成するとともに、この第二の共有鍵、上記第一の乱数および上記第二の乱数を引数とする一方向関数の関数値を第三のワンタイムＩＤとして求め、この第三のワンタイムＩＤを上記第二装置に対して送信するステップと、

上記第二装置が、上記第一の乱数および上記第二の乱数に基づいて上記第二の共有鍵を生成するとともに、この第二の共有鍵、上記第一の乱数および上記第二の乱数に基づいて上記第三のワンタイムＩＤを演算により求め、この演算結果と上記第一装置から受信した上記第三のワンタイムＩＤとの比較により、上記第一装置の正当性を判定するステップとを有することを特徴とする認証方法。

【請求項１２】

請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いて、第一装置と第二装置間における認証を行う認証方法であって、

上記第一装置が、第一の乱数を生成するとともに、上記第二装置との間で予め共有化された共有鍵を引数とする一方向関数の関数値を第一のワンタイムＩＤとして求め、この第一のワンタイムＩＤと上記第一の乱数を上記第二装置に対して送信するステップと、

上記第二装置が、第二の乱数を生成するとともに、上記第一の乱数と上記共有鍵とを引数とする一方向関数の関数値を第二のワンタイムＩＤとして求め、この第二のワンタイムＩＤと上記第二の乱数を上記第一装置に対して送信するステップと、

上記第一装置が、上記第一の乱数および上記共有鍵に基づいて上記第二のワンタイムＩＤを演算により求め、この演算結果と上記第二装置から受信した上記第二のワンタイムＩＤとの比較により、上記第二装置の正当性を判定するステップと、

上記第一装置が、上記第一の乱数、上記第二の乱数および上記共有鍵を引数とする一方向関数の関数値を第三のワンタイムＩＤとして求め、この第三のワンタ

イムＩＤを上記第二装置に対して送信するステップと、

上記第二装置が、上記第一の乱数、上記第二の乱数および上記共有鍵に基づいて上記第三のワンタイムＩＤを演算により求め、この演算結果と上記第一装置から受信した上記第三のワンタイムＩＤとの比較により、上記第一装置の正当性を判定するステップとを有することを特徴とする認証方法。

【請求項１３】

上記第一の乱数と上記第二の乱数を、上記第一装置と上記第二装置との間で予め共有化された共有鍵で暗号化した状態で、送信するようにしたことを特徴とする請求項１１または請求項１２に記載の認証方法。

【請求項１４】

上記第二装置が上記第二のワンタイムＩＤと上記第二の乱数とを上記第一装置に対して送信するステップにおいて、上記第二装置は、上記第一装置との間で予め共有化された乱数を初期乱数として、この初期乱数と上記第一の乱数を引数とする所定の演算を行い、この演算結果を上記第一装置に対して送信する一方、上記第一装置は、上記第二装置の正当性の判定材料として、上記第二装置から受信した上記演算結果を、上記第二のワンタイムＩＤとともに用いることを特徴とする請求項１１～請求項１３の何れかに記載の認証方法。

【請求項１５】

上記第一装置が上記第三のワンタイムＩＤを上記第二装置に対して送信するステップにおいて、上記第一装置は、上記第一の乱数と上記第二の乱数を引数とする所定の演算を行い、この演算結果を上記第二装置に対して送信する一方、上記第二装置は、上記第一装置の正当性の判定材料として、上記第一装置から受信した上記演算結果を、上記第三のワンタイムＩＤとともに用いることを特徴とする請求項１１～請求項１４の何れかに記載の認証方法。

【請求項１６】

請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いて、第一装置と第二装置間における認証を行う認証方法であって、

上記第一装置が、第一の乱数を生成するとともに、上記第二装置との間で予め

共有化された共有鍵、第一の記憶乱数および第二の記憶乱数を引数とする一方向関数の関数値を第一のワンタイムＩＤとして求め、当該第一装置に予め設定されたＩＤ、上記第二装置に予め設定されたＩＤおよび上記第一の乱数を上記共有鍵で暗号化した第一の暗号化データと、上記第一のワンタイムＩＤとを上記第二装置に対して送信するステップと、

上記第二装置が、上記第一のワンタイムＩＤを演算により求め、この演算結果と、上記第一装置から受信した上記第一のワンタイムＩＤとの照合により、上記第一装置を識別するステップと、

上記第二装置が、上記第一装置を識別できた場合に、上記共有鍵を用いて上記第一の暗号化データを復号し、この復号したデータに含まれる、上記第一装置に予め設定されたＩＤおよび当該第二装置に予め設定されたＩＤに基づいて、上記第一装置の正当性を判定するステップと、

上記第二装置が、上記第一装置を正当であると判定した場合に、第二の乱数を生成するとともに、上記第一の乱数、上記第二の記憶乱数および上記共有鍵を引数とする一方向関数の関数値を第二のワンタイムＩＤとして求め、上記第一装置に予め設定されたＩＤ、当該第二装置に予め設定されたＩＤおよび上記第二の乱数を上記共有鍵で暗号化した第二の暗号化データと、上記第二のワンタイムＩＤとを上記第一装置に対して送信するステップと、

上記第二装置が、上記第一の記憶乱数を上記第一の乱数に、上記第二の記憶乱数を上記第二の乱数にそれぞれ置換するステップと、

上記第一装置が、上記第二のワンタイムＩＤを演算により求め、この演算結果と、上記第二装置から受信した上記第二のワンタイムＩＤとの照合により、上記第二装置を識別するステップと、

上記第一装置が、上記第二装置を識別できた場合に、上記共有鍵を用いて上記第二の暗号化データを復号し、この復号したデータに含まれる、上記第二装置に予め設定されたＩＤおよび当該第一装置に予め設定されたＩＤに基づいて、上記第二装置の正当性を判定するステップと、

上記第一装置が、上記第一の記憶乱数を上記第一の乱数に、上記第二の記憶乱数を上記第二の乱数にそれぞれ置換するステップとを有することを特徴とする認

証方法。

【請求項１７】

上記第一の記憶乱数を上記第一の乱数に、上記第二の記憶乱数を上記第二の乱数にそれぞれ置換した後に、これら第一の記憶乱数および第二の記憶乱数に基づいて上記共有鍵を生成することにより、当該共有鍵を変化させるようにしたことを特徴とする請求項１６に記載の認証方法。

【請求項１８】

請求項１または請求項４に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてクライアントとの間で認証を行うサーバであって、

上記クライアントに予め設定されたクライアントＩＤを少なくとも引数とする一方向関数Ｆｃの関数値と、上記クライアントに予め記憶されたDiffie-Hellman公開値の一方と、上記ワンタイムＩＤとを上記クライアントから受信する受信手段と、

上記一方向関数の関数値Ｆｃおよび上記ワンタイムＩＤを演算により求め、この演算結果と、上記クライアントから受信した上記ワンタイムＩＤおよび上記一方向関数Ｆｃの関数値との比較により、上記クライアントの正当性を判定する判定手段と、

上記判定手段が上記クライアントを正当であると判定した場合に、当該サーバに予め設定されたサーバＩＤを少なくとも引数とする一方向関数Ｆｓの関数値と、当該サーバに予め記憶されたDiffie-Hellman公開値の他方とを上記クライアントに対して送信する送信手段とを備えることを特徴とするサーバ。

【請求項１９】

請求項１または請求項４に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてサーバとの間で認証を行うクライアントであって、

上記サーバとの間で予め共有化された可変共有鍵を用いて上記ワンタイムＩＤを生成するとともに、当該クライアントに予め設定されたクライアントＩＤを少なくとも引数とする一方向関数Ｆｃの関数値を演算により求め、これらワンタイムＩＤおよび一方向関数Ｆｃの関数値と、当該クライアントに予め記憶されたDiffie-Hellman公開値の一方とを上記サーバに対して送信する送信手段と、

上記サーバに予め設定されたサーバＩＤを少なくとも引数とする一方向関数Ｆｓの関数値と、上記サーバに予め記憶されたDiffie-Hellman公開値の他方とを上記サーバから受信する受信手段と、

上記一方向関数Ｆｓの関数値を演算により求め、この演算結果と、上記サーバから受信した上記一方向関数Ｆｓの関数値との比較により、上記サーバの正当性を判定する判定手段とを備えることを特徴とするクライアント。

【請求項２０】

請求項１８に記載のサーバと、請求項１９に記載のクライアントとを備えてなることを特徴とする認証システム。

【請求項２１】

請求項１または請求項４に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤに基づいてクライアントとの間で認証を行うサーバに実行させるプログラムであって、

上記クライアントに予め設定されたクライアントＩＤを少なくとも引数とする一方向関数Ｆｃの関数値と、上記クライアントに予め記憶されたDiffie-Hellman公開値の一方と、上記ワンタイムＩＤとを上記クライアントから受信する処理と、

上記一方向関数の関数値Ｆｃおよび上記ワンタイムＩＤを演算により求め、この演算結果と、上記クライアントから受信した上記ワンタイムＩＤおよび上記一方向関数Ｆｃの関数値との比較により、上記クライアントの正当性を判定する処理と、

上記クライアントが正当であると判定された場合に、上記サーバに予め設定されたサーバＩＤを少なくとも引数とする一方向関数Ｆｓの関数値と、上記サーバに予め記憶されたDiffie-Hellman公開値の他方とを上記クライアントに対して送信する処理とを上記サーバに実行させることを特徴とするプログラム。

【請求項２２】

請求項１または請求項４に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤに基づいてサーバとの間で認証を行うクライアントに実行させるプログラムであって、

上記サーバとの間で予め共有化された可変共有鍵を用いて上記ワンタイムＩＤを生成するとともに、上記クライアントに予め設定されたクライアントＩＤを少なくとも引数とする一方向関数Ｆｃの関数値を演算により求め、これらワンタイムＩＤおよび一方向関数Ｆｃの関数値と、上記クライアントに予め記憶されたDiffie-Hellman公開値の一方とを上記サーバに対して送信する処理と、

上記サーバに予め設定されたサーバＩＤを少なくとも引数とする一方向関数Ｆｓの関数値と、上記サーバに予め記憶されたDiffie-Hellman公開値の他方とを上記サーバから受信する処理と、

上記一方向関数Ｆｓの関数値を演算により求め、この演算結果と、上記サーバから受信した上記一方向関数Ｆｓの関数値との比較により、上記サーバの正当性を判定する処理とを上記クライアントに実行させることを特徴とするプログラム。

【請求項２３】

請求項２または請求項５に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてクライアントとの間で認証を行うサーバであって、

上記クライアントとの間で予め共有化された第一の可変共有鍵と上記クライアントの通信順序に関する情報とを引数とする一方向関数の関数値を第一のワンタイムＩＤとして、この第一のワンタイムＩＤ、上記クライアントに予め設定されたクライアントＩＤ、当該サーバに予め設定されたサーバＩＤ、上記クライアントに予め記憶されたDiffie-Hellman公開値の一方を上記第一の可変共有鍵で暗号化した暗号化データと、上記第一のワンタイムＩＤとを上記クライアントから受信する受信手段と、

上記第一のワンタイムＩＤを演算により求め、この演算結果と、上記クライアントから受信した上記第一のワンタイムＩＤとの照合により、上記クライアントを識別し、上記クライアントを識別できた場合に、上記第一の可変共有鍵を用いて上記暗号化データを復号し、この復号したデータに含まれる、上記クライアントＩＤ、上記サーバＩＤおよび上記第一のワンタイムＩＤに基づいて、上記クライアントの正当性を判定する判定手段と、

上記判定手段が上記クライアントを正当であると判定した場合に、上記第一の

可変共有鍵と当該サーバの通信順序に関する情報とを引数とする一方向関数の関数値を第二のワンタイムＩＤとして生成するとともに、上記クライアントから受信したDiffie-Hellman公開値の一方と当該サーバに予め記憶されたDiffie-Hellman公開値の他方とからDiffie-Hellman共通鍵を第二の可変共有鍵として生成し、この第二の可変共有鍵、上記クライアントＩＤ、上記サーバＩＤおよび上記第二のワンタイムＩＤを引数とする一方向関数ｈの関数値と、上記Diffie-Hellman公開値の他方と、上記第二のワンタイムＩＤとを上記クライアントに対して送信する送信手段とを備えることを特徴とするサーバ。

【請求項２４】

請求項２または請求項５に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてサーバとの間で認証を行うクライアントであって、

上記サーバとの間で予め共有化された第一の可変共有鍵と当該クライアントの通信順序に関する情報とを引数とする一方向関数の関数値を第一のワンタイムＩＤとして生成するとともに、上記第一の可変共有鍵を用いて、当該クライアントに予め設定されたクライアントＩＤ、上記サーバに予め設定されたサーバＩＤ、当該クライアントに予め記憶されたDiffie-Hellman公開値の一方および上記第一のワンタイムＩＤを暗号化し、この暗号化データと上記第一のワンタイムＩＤとを上記サーバに対して送信する送信手段と、

上記第一の可変共有鍵と上記サーバの通信順序に関する情報とを引数とする一方向関数の関数値を第二のワンタイムＩＤとし、Diffie-Hellman共通鍵を第二の可変共有鍵として、上記第二のワンタイムＩＤ、上記第二の可変共有鍵、上記クライアントＩＤおよび上記サーバＩＤを引数とする一方向関数ｈの関数値と、上記サーバに予め記憶されたDiffie-Hellman公開値の他方と、上記第二のワンタイムＩＤとを上記サーバから受信する受信手段と、

上記第二のワンタイムＩＤを演算により求め、この演算結果と、上記サーバから受信した上記第二のワンタイムＩＤとの照合により、上記サーバを識別し、上記サーバを識別した場合に、上記サーバから受信した上記Diffie-Hellman公開値の他方と当該クライアントに予め記憶された上記Diffie-Hellman公開値の一方とからDiffie-Hellman共通鍵を上記第二の可変共有鍵として生成するとともに、こ

の第二の可変共有鍵を用いて上記一方向関数ｈの関数値を演算により求め、この演算結果と、上記サーバから受信した一方向関数ｈの関数値との照合により、上記サーバの正当性を判定する判定手段とを備えることを特徴とするクライアント。

【請求項２５】

請求項２３に記載のサーバと、請求項２４に記載のクライアントとを備えてなることを特徴とする認証システム。

【請求項２６】

請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてクライアントとの間で相互に認証を行うサーバであって、

上記クライアントとの間で予め共有化された第一の共有鍵を引数とする一方向関数の関数値を第一のワンタイムＩＤとして、この第一のワンタイムＩＤと、上記クライアントで生成された第一の乱数とを上記クライアントから受信する第一受信手段と、

第二の乱数を生成するとともに、上記第一の乱数と上記第一の共有鍵とを引数とする一方向関数の関数値を第二のワンタイムＩＤとして求め、この第二のワンタイムＩＤと上記第二の乱数とを上記クライアントに対して送信する送信手段と、

上記第一の乱数、上記第二の乱数および第二の共有鍵を引数とする一方向関数の関数値を第三のワンタイムＩＤとして、この第三のワンタイムＩＤを上記クライアントから受信する第二受信手段と、

上記第一の乱数および上記第二の乱数に基づいて上記第二の共有鍵を生成するとともに、この第二の共有鍵、上記第一の乱数および上記第二の乱数に基づいて上記第三のワンタイムＩＤを演算により求め、この演算結果と上記クライアントから受信した上記第三のワンタイムＩＤとの比較により、上記クライアントの正当性を判定する判定手段とを備えることを特徴とするサーバ。

【請求項２７】

請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成された

ワンタイムＩＤを用いてサーバとの間で相互に認証を行うクライアントであって、

第一の乱数を生成するとともに、上記サーバとの間で予め共有化された第一の共有鍵を引数とする一方向関数の関数値を第一のワンタイムＩＤとして求め、この第一のワンタイムＩＤと上記第一の乱数とを上記サーバに対して送信する第一送信手段と、

上記第一の乱数と上記第一の共有鍵とを引数とする一方向関数の関数値を第二のワンタイムＩＤとして、この第二のワンタイムＩＤと、上記サーバで生成された第二の乱数とを上記サーバから受信する受信手段と、

上記第一の乱数および上記第一の共有鍵に基づいて上記第二のワンタイムＩＤを演算により求め、この演算結果と上記サーバから受信した上記第二のワンタイムＩＤとの比較により、上記サーバの正当性を判定する判定手段と、

上記判定手段により上記サーバが正当であると判定された場合に、上記第一の乱数および上記第二の乱数に基づいて第二の共有鍵を生成するとともに、この第二の共有鍵、上記第一の乱数および上記第二の乱数を引数とする一方向関数の関数値を第三のワンタイムＩＤとして求め、この第三のワンタイムＩＤを上記サーバに対して送信する第二送信手段とを備えることを特徴とするクライアント。

【請求項２８】

請求項２６に記載のサーバと、請求項２７に記載のクライアントとを備えてなることを特徴とする認証システム。

【請求項２９】

請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてクライアントとの間で相互に認証を行うサーバであって、

上記クライアントとの間で予め共有化された共有鍵を引数とする一方向関数の関数値を第一のワンタイムＩＤとして、この第一のワンタイムＩＤと、上記クライアントで生成された第一の乱数とを上記クライアントから受信する第一受信手段と、

第二の乱数を生成するとともに、上記第一の乱数と上記共有鍵とを引数とする

一方向関数の関数値を第二のワンタイムＩＤとして求め、この第二のワンタイムＩＤと上記第二の乱数を上記クライアントに対して送信する送信手段と、

上記共有鍵、上記第一の乱数および上記第二の乱数を引数とする一方向関数の関数値を第三のワンタイムＩＤとして、この第三のワンタイムＩＤを上記クライアントから受信する第二受信手段と、

上記第一の乱数、上記第二の乱数および上記共有鍵に基づいて上記第三のワンタイムＩＤを演算により求め、この演算結果と上記クライアントから受信した上記第三のワンタイムＩＤとの比較により、上記クライアントの正当性を判定する判定手段とを備えることを特徴とするサーバ。

【請求項３０】

請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてサーバとの間で相互に認証を行うクライアントであって、

第一の乱数を生成するとともに、上記サーバとの間で予め共有化された共有鍵を引数とする一方向関数の関数値を第一のワンタイムＩＤとして求め、この第一のワンタイムＩＤと上記第一の乱数を上記サーバに対して送信する第一送信手段と、

上記第一の乱数と上記共有鍵とを引数とする一方向関数の関数値を第二のワンタイムＩＤとして、この第二のワンタイムＩＤと、上記サーバで生成された第二の乱数とを上記サーバから受信する受信手段と、

上記第一の乱数および上記共有鍵に基づいて上記第二のワンタイムＩＤを演算により求め、この演算結果と上記サーバから受信した上記第二のワンタイムＩＤとの比較により、上記サーバの正当性を判定する判定手段と、

上記判定手段により上記サーバが正当であると判定された場合に、上記第一の乱数、上記第二の乱数および上記共有鍵を引数とする一方向関数の関数値を第三のワンタイムＩＤとして求め、この第三のワンタイムＩＤを上記サーバに対して送信する第二送信手段とを備えることを特徴とするクライアント。

【請求項３１】

請求項２９に記載のサーバと、請求項３０に記載のクライアントとを備えてな

ることを特徴とする認証システム。

【請求項３２】

請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてクライアントとの間で相互に認証を行うサーバであって、

上記クライアントとの間で予め共有化された共有鍵、第一の記憶乱数および第二の記憶乱数を引数とする一方向関数の関数値を第一のワンタイムＩＤとして、この第一のワンタイムＩＤを上記クライアントから受信するとともに、上記クライアントで生成された第一の乱数、上記クライアントに予め設定されたクライアントＩＤ、当該サーバに予め設定されたサーバＩＤを上記共有鍵で暗号化した第一の暗号化データを上記クライアントから受信する受信手段と、

上記第一のワンタイムＩＤを演算により求め、この演算結果と、上記クライアントから受信した上記第一のワンタイムＩＤとの照合により、上記クライアントを識別し、上記クライアントを識別できた場合に、上記共有鍵を用いて上記第一の暗号化データを復号し、この復号したデータに含まれる上記クライアントＩＤおよび上記サーバＩＤに基づいて、上記クライアントの正当性を判定する判定手段と、

上記判定手段が上記クライアントを正当であると判定した場合に、第二の乱数を生成するとともに、上記第一の乱数、上記第二の記憶乱数および上記共有鍵を引数とする一方向関数の関数値を第二のワンタイムＩＤとして求め、上記クライアントＩＤ、上記サーバＩＤおよび上記第二の乱数を上記共有鍵で暗号化した第二の暗号化データと、上記第二のワンタイムＩＤとを上記クライアントに対して送信する送信手段と、

上記第一の記憶乱数を上記第一の乱数に、上記第二の記憶乱数を上記第二の乱数にそれぞれ置換する置換手段とを備えることを特徴とするサーバ。

【請求項３３】

請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてサーバとの間で相互に認証を行うクライアントであって、

第一の乱数を生成するとともに、上記サーバとの間で予め共有化された共有鍵、第一の記憶乱数および第二の記憶乱数を引数とする一方向関数の関数値を第一のワンタイムＩＤとして求め、当該クライアントに予め設定されたクライアントＩＤ、上記サーバに予め設定されたサーバＩＤおよび上記第一の乱数を上記共有鍵で暗号化した第一の暗号化データと、上記第一のワンタイムＩＤとを上記サーバに対して送信する送信手段と、

上記第一の乱数、上記第二の記憶乱数および上記共有鍵を引数とする一方向関数の関数値を第二のワンタイムＩＤとして、この第二のワンタイムＩＤを上記サーバから受信するとともに、上記サーバで生成された第二の乱数、上記クライアントＩＤおよび上記サーバＩＤを上記共有鍵で暗号化した第二の暗号化データを上記サーバから受信する受信手段と、

上記第二のワンタイムＩＤを演算により求め、この演算結果と、上記サーバから受信した上記第二のワンタイムＩＤとの照合により、上記サーバを識別し、上記サーバを識別できた場合に、上記共有鍵を用いて上記第二の暗号化データを復号し、この復号したデータに含まれる上記サーバＩＤおよび上記クライアントＩＤに基づいて、上記サーバの正当性を判定する判定手段と、

上記第一の記憶乱数を上記第一の乱数に、上記第二の記憶乱数を上記第二の乱数にそれぞれ置換する置換手段とを備えることを特徴とするクライアント。

【請求項３４】

請求項３２に記載のサーバと、請求項３３に記載のクライアントとを備えてなることを特徴とする認証システム。

【請求項３５】

上記サーバおよび上記クライアントは、上記第一の記憶乱数を上記第一の乱数に、上記第二の記憶乱数を上記第二の乱数にそれぞれ置換した後で、これら第一の記憶乱数および第二の記憶乱数に基づいて上記共有鍵を生成することにより、当該共有鍵を変化させるようになっていることを特徴とする請求項３４に記載の認証システム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、複数の装置間またはアプリケーション間における認証に用いて好適なワンタイムＩＤの生成方法、上記ワンタイムＩＤを用いた認証方法、認証システム、サーバ、クライアントおよびプログラムに関するものである。

【０００２】

【従来の技術】

従来より、ネットワークを介してコンピュータ間（例えば、クライアント・サーバ間）で通信を行う際には、不正なアクセス等を排除するために、サービス等の提供に先立って認証が行われる。この認証においては、第三者が知り得ない所定の秘密情報（例えば、ＩＤやパスワード、乱数、或いはそれら情報を引数とする関数値など）を予め双方が共有し、その秘密情報に基づいて各々の正当性を相互に検証するのが一般的である。

【０００３】

他方、ＩＥＴＦ（Internet Engineering Task Force）が公式に発行するＲＦＣ（Request For Comments）においては、インターネットでＩＰパケットの暗号化と認証を行なうセキュリティプロトコルとして、ＩＰｓｅｃ（Security Architecture for Internet Protocol）が規定されている。このＩＰｓｅｃでは、暗号・認証のパラメータを動的に生成して交換するＩＫＥ(Internet Key Exchange)という自動鍵交換のプロトコルが標準的に採用されている（例えば、特許文献１参照）。

そして、近年では、このＩＫＥの方式にワンタイムＩＤを導入して、既知共有鍵を用いたＩＫＥの方式で問題となっていた、ＩＤ情報保護、ＤｏＳ（Denial of Service attack）攻撃防止、リモートアクセスなどを実現したＰ－ＳＩＧＭＡと呼ばれる鍵交換・認証方式が提案されている。

【０００４】

このＰ－ＳＩＧＭＡにおいては、例えば、図１３に示すような手順で鍵交換および認証が行われている。

先ず、クライアントが、ＳＡ（Security Association）の提案、乱数Ｒｃ、ＤＨ（Diffie-Hellman）公開値g^x、ＯＩＤ（ワンタイムＩＤ）をサーバに対して

送信する。なお、ＳＡの提案には、暗号アルゴリズムや認証方式、鍵交換に使用するパラメータ等に関する提案が含まれている。

【０００５】

次いで、サーバが、受け取ったＯＩＤからクライアントを識別し、識別できない場合には、通信を拒否する。識別できる場合には、受諾したＳＡ、乱数Ｒｓ、ＤＨ公開値g^y、ＨＡＳＨｓ、セッション鍵ｅで暗号化したＩＤｓ（サーバＩＤ）をクライアントに対して送信する。なお、セッション鍵ｅは、既知共有鍵、乱数Ｒｓ、乱数ＲｃおよびＤＨ共通鍵g^{xy}を引数とする鍵付きハッシュ関数の関数値であり、ＨＡＳＨｓは、既知共有鍵、乱数Ｒｓ、乱数Ｒｃ、ＤＨ公開値g^x、g^yおよびＩＤｓを引数とする疑似乱数関数の関数値である。

【０００６】

次いで、クライアントが、受け取ったＨＡＳＨｓを検証し、このＨＡＳＨｓに基づいてサーバの正当性を確認する。ＨＡＳＨｓが正しければ、ＨＡＳＨｃ、セッション鍵ｅで暗号化したＩＤｃ（クライアントＩＤ）をサーバに対して送信する。ここで、ＨＡＳＨｃは、既知共有鍵、乱数Ｒｓ、乱数Ｒｃ、ＤＨ公開値g^x、g^yおよびＩＤｃを引数とする疑似乱数関数の関数値である。

次いで、サーバが、受け取ったＨＡＳＨｃを検証し、このＨＡＳＨｃに基づいてクライアントの正当性を確認する。ＨＡＳＨｃが正しければ、当該プロトコルを終了する。

【０００７】

このＰ－ＳＩＧＭＡにおいて、ＯＩＤ（ワンタイムＩＤ）は、次のように定義されている。

【０００８】

【数１】

$$OID_1 = prf(K, 1)$$
$$OID_2 = prf(K, 2)$$
$$\cdots$$
$$OID_n = prf(K, n)$$

【０００９】

この定義式において、OID_n はｎ番目のＳＡ確立時に用いられるワンタイムＩＤ、ｐｒｆは疑似乱数関数、Ｋは既知共有鍵、若しくは既知共有鍵から生成された値である。

【００１０】

このため、上記Ｐ－ＳＩＧＭＡによれば、ＯＩＤを導入したことにより、第三者が送信者・受信者を特定できなくなる一方で、正当な送信者・受信者であればＯＩＤを識別情報として把握できるといった効果が得られるとともに、クライアント・サーバ間で通信が行われる度（すなわち、ＳＡの生成または更新毎）にＯＩＤが変更されるため、第三者が次回のＯＩＤを予測することができないといった効果が得られる。

【００１１】

【特許文献１】

特開２００２－３７４２３８号公報（段落番号０００２～０００９）

【００１２】

【発明が解決しようとする課題】

しかしながら、上記Ｐ－ＳＩＧＭＡにおいては、既知共有鍵がひとたび知られてしまうと、すべてのＯＩＤが予測されてしまい、その結果、ＯＩＤの将来にわたる安全性（すなわち、ＰＦＳ：Perfect Forward Security）を保証できなくなるといった問題点があった。

【００１３】

以上、具体例として、Ｐ－ＳＩＧＭＡと呼ばれる鍵交換・認証方式について述べてきたが、一般に、ワンタイムＩＤを用いて複数の装置間またはアプリケーション間における認証を行う認証方式では、特定の秘密情報に基づいてすべてのワンタイムＩＤの生成を行っており、上記同様の問題点を有している。

【００１４】

本発明は、かかる事情に鑑みてなされたもので、盗聴が困難で安全性に優れたワンタイムＩＤの生成方法、上記ワンタイムＩＤを用いた認証方法、認証システム、サーバ、クライアントおよびプログラムを提供することを目的とする。

【００１５】

【課題を解決するための手段】

請求項１に記載の発明は、複数の装置間またはアプリケーション間における認証において一回限り使用可能な識別情報をワンタイムＩＤとして、当該ワンタイムＩＤを生成する方法であって、上記認証を行う装置またはアプリケーションの各々において、上記認証が必要な所定の通信単位毎に変化する可変共有鍵を生成するとともに、この可変共有鍵を引数とする一方向関数の関数値を求め、この関数値から上記ワンタイムＩＤを生成するようにしたことを特徴とするものである。

【００１６】

ここで、一方向関数とは、引数から結果（関数値）を求めるのは簡単であるが、結果から引数を求めるのは難しい関数のことを云い、この一方向関数には、例えば、ハッシュ関数、疑似乱数関数などが含まれる。

所定の通信単位としては、例えば、ＩＰｓｅｃにおいてＳＡが確立されてから当該ＳＡが無効になるまでの間にクライアント・サーバ間で行われる一連の通信を、所定の通信単位として設定することも可能であるし、装置間またはアプリケーション間で行われる１回のデータ送受信を所定の通信単位として設定することも可能である。

可変共有鍵は、上記所定の通信単位毎に変化し、且つ認証を行う装置間またはアプリケーション間で共有される、第三者が知り得ない秘密情報であれば、如何なる鍵であってもよい。

【００１７】

認証とは、一方の装置（または一方のアプリケーション）が他方の装置（または他方のアプリケーション）にアクセスする際に、他方の装置が一方の装置の正当性を確認することを云い、識別情報とは、上記認証において少なくとも一方の装置から他方の装置に送信されて当該他方の装置が一方の装置を識別するのに用いる情報（ＩＤ）のことを云う。

また、上記認証には、一方の装置が他方の装置の認証を行う一方向認証と、双方の装置で相互に認証を行う相互認証とが含まれる。例えば、上記認証においてワンタイムＩＤを使用する方法としては、双方の装置でワンタイムＩＤを生成す

るとともに、一方の装置が他方の装置にワンタイムＩＤを送信し、他方の装置が、一方の装置から受信したワンタイムＩＤと自らが生成したワンタイムＩＤとの比較・照合により、一方の装置を識別或いは認証する方法が挙げられる。

【００１８】

請求項２に記載の発明は、複数の装置間またはアプリケーション間における認証において一回限り使用可能な識別情報をワンタイムＩＤとして、当該ワンタイムＩＤを生成する方法であって、上記認証を行う装置またはアプリケーションの各々において、上記認証が必要な所定の通信単位毎に変化する可変共有鍵を生成するとともに、この可変共有鍵と通信順序または回数に関する情報とを引数とする一方向関数の関数値を求め、この関数値から上記ワンタイムＩＤを生成するようにしたことを特徴とするものである。

【００１９】

請求項３に記載の発明は、複数の装置間またはアプリケーション間における認証において一回限り使用可能な識別情報をワンタイムＩＤとして、当該ワンタイムＩＤを生成する方法であって、上記認証を行う装置またはアプリケーションの各々において、上記認証が必要な所定の通信単位内で乱数を生成するとともに、この乱数と所定の共有鍵とを引数とする一方向関数の関数値を求め、この関数値から上記ワンタイムＩＤを生成するようにしたことを特徴とするものである。

【００２０】

請求項４に記載の発明は、一方の装置と他方の装置間における認証において一回限り使用可能な識別情報をワンタイムＩＤとして、当該ワンタイムＩＤを双方の装置で生成するとともに、一方の装置が他方の装置にワンタイムＩＤを送信して、他方の装置が、一方の装置から受信したワンタイムＩＤと自らが生成したワンタイムＩＤとの比較・照合により、他方の装置を識別或いは認証する場合において、一方の装置および他方の装置がワンタイムＩＤを生成する方法であって、一方の装置および他方の装置は、上記認証が必要な所定の通信単位毎に変化する可変共有鍵を生成するとともに、この可変共有鍵を引数とする一方向関数の関数値を求め、この関数値から上記ワンタイムＩＤを生成するようにしたことを特徴とするものである。

【００２１】

請求項５に記載の発明は、一方の装置と他方の装置間における認証において一回限り使用可能な識別情報をワンタイムＩＤとして、当該ワンタイムＩＤを双方の装置で生成するとともに、一方の装置が他方の装置にワンタイムＩＤを送信して、他方の装置が、一方の装置から受信したワンタイムＩＤと自らが生成したワンタイムＩＤとの比較・照合により、他方の装置を識別或いは認証する場合において、一方の装置および他方の装置がワンタイムＩＤを生成する方法であって、一方の装置および他方の装置は、上記認証が必要な所定の通信単位毎に変化する可変共有鍵を生成するとともに、この可変共有鍵と通信順序または回数に関する情報とを引数とする一方向関数の関数値を求め、この関数値から上記ワンタイムＩＤを生成するようにしたことを特徴とするものである。

【００２２】

請求項６に記載の発明は、一方の装置と他方の装置間における認証において一回限り使用可能な識別情報をワンタイムＩＤとして、当該ワンタイムＩＤを双方の装置で生成するとともに、一方の装置が他方の装置にワンタイムＩＤを送信して、他方の装置が、一方の装置から受信したワンタイムＩＤと自らが生成したワンタイムＩＤとの比較・照合により、他方の装置を識別或いは認証する場合において、一方の装置および他方の装置がワンタイムＩＤを生成する方法であって、一方の装置および他方の装置は、上記認証が必要な所定の通信単位内で乱数を生成するとともに、この乱数と所定の共有鍵とを引数とする一方向関数の関数値を求め、この関数値から上記ワンタイムＩＤを生成するようにしたことを特徴とするものである。

【００２３】

請求項７に記載の発明は、請求項１または請求項４に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤ（$SIGNAL_n$）を用いて、第一装置と第二装置間における認証を行う認証方法であって、上記第一装置が、上記第二装置との間で予め共有化された可変共有鍵を用いて上記ワンタイムＩＤを生成するとともに、この生成したワンタイムＩＤと、当該第一装置に予め設定されたＩＤを少なくとも引数とする一方向関数Ｆｃの関数値と、当該第一装置に予め記

憶されたDiffie-Hellman公開値の一方とを上記第二装置に対して送信するステップと、上記第二装置が、上記ワンタイムＩＤおよび上記一方向関数Ｆｃの関数値を演算により求め、この演算結果と、上記第一装置から受信したワンタイムＩＤおよび一方向関数Ｆｃの関数値との照合により、上記第一装置の正当性を判定するステップと、上記第二装置が、上記第一装置を正当であると判定した場合に、当該第二装置に予め設定されたＩＤを少なくとも引数とする一方向関数Ｆｓの関数値と、当該第二装置に予め記憶されたDiffie-Hellman公開値の他方とを上記第一装置に対して送信するステップと、上記第一装置が、上記一方向関数Ｆｓの関数値を演算により求め、この演算結果と、上記第二装置から受信した一方向関数Ｆｓの関数値との照合により、上記第二装置の正当性を判定するステップとを有することを特徴とするものである。

【００２４】

請求項８に記載の発明は、請求項７に記載の認証方法において、上記一方向関数Ｆｃとして、所定の共有鍵、上記Diffie-Hellman公開値の一方、上記第一装置に予め設定されたＩＤ、上記ワンタイムＩＤを引数とする疑似乱数関数を用いるとともに、上記一方向関数Ｆｓとして、上記所定の共有鍵、上記Diffie-Hellman公開値の一方、上記Diffie-Hellman公開値の他方、上記第二装置に予め設定されたＩＤ、上記ワンタイムＩＤを引数とする疑似乱数関数を用いるようにしたことを特徴とするものである。

【００２５】

請求項９に記載の発明は、請求項２または請求項５に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いて、第一装置と第二装置間における認証を行う認証方法であって、上記第一装置が、上記第二装置との間で予め共有化された第一の可変共有鍵と当該第一装置の通信順序に関する情報とを引数とする一方向関数の関数値を第一のワンタイムＩＤ（$SIGNAL_{n,j}$）として生成するとともに、上記第一の可変共有鍵を用いて、当該第一装置に予め設定されたＩＤ、上記第二装置に予め設定されたＩＤ、当該第一装置に予め記憶されたDiffie-Hellman公開値の一方および上記第一のワンタイムＩＤを暗号化し、この暗号化データと上記第一のワンタイムＩＤとを上記第二装置に対して送信するス

テップと、上記第二装置が、上記第一のワンタイムＩＤを演算により求め、この演算結果と、上記第一装置から受信した上記第一のワンタイムＩＤとの照合により、上記第一装置を識別するステップと、上記第二装置が、上記第一装置を識別できた場合に、上記第一の可変共有鍵を用いて上記暗号化データを復号し、この復号したデータに含まれる、上記第一装置に予め設定されたＩＤ、当該第二装置に予め設定されたＩＤおよび上記第一のワンタイムＩＤに基づいて、上記第一装置の正当性を判定するステップと、上記第二装置が、上記第一装置を正当であると判定した場合に、上記第一の可変共有鍵と当該第二装置の通信順序に関する情報とを引数とする一方向関数の関数値を第二のワンタイムＩＤ（ＳＩＧＮＡＬ’$_{n,1}$）として生成するとともに、上記第一装置から受信したDiffie-Hellman公開値の一方と当該第二装置に予め記憶されたDiffie-Hellman公開値の他方とからDiffie-Hellman共通鍵を第二の可変共有鍵として生成し、この第二の可変共有鍵、上記第一装置に予め設定されたＩＤ、当該第二装置に予め設定されたＩＤおよび上記第二のワンタイムＩＤを引数とする一方向関数ｈの関数値と、上記Diffie-Hellman公開値の他方と、上記第二のワンタイムＩＤとを上記第一装置に対して送信するステップと、上記第一装置が、上記第二のワンタイムＩＤを演算により求め、この演算結果と、上記第二装置から受信した上記第二のワンタイムＩＤとの照合により、上記第二装置を識別するステップと、上記第一装置が、上記第二装置を識別できた場合に、上記第二装置から受信した上記Diffie-Hellman公開値の他方と当該第一装置に予め記憶された上記Diffie-Hellman公開値の一方とからDiffie-Hellman共通鍵を上記第二の可変共有鍵として生成するとともに、この第二の可変共有鍵を用いて上記一方向関数ｈの関数値を演算により求め、この演算結果と、上記第二装置から受信した一方向関数ｈの関数値との照合により、上記第二装置の正当性を判定するステップとを有することを特徴とするものである。

【００２６】

請求項１０に記載の発明は、請求項９に記載の認証方法において、上記第二のワンタイムＩＤを生成する一方向関数として、上記第一のワンタイムＩＤを生成する一方向関数とは異なる一方向関数を用いるようにしたことを特徴とするものである。

【００２７】

請求項１１に記載の発明は、請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いて、第一装置と第二装置間における認証（相互認証）を行う認証方法であって、上記第一装置が、第一の乱数を生成するとともに、上記第二装置との間で予め共有化された第一の共有鍵を引数とする一方向関数の関数値を第一のワンタイムＩＤ（$SIGNAL_{c1}$）として求め、この第一のワンタイムＩＤと上記第一の乱数とを上記第二装置に対して送信するステップと、上記第二装置が、第二の乱数を生成するとともに、上記第一の乱数と上記第一の共有鍵とを引数とする一方向関数の関数値を第二のワンタイムＩＤ（$SIGNAL_{s1}$）として求め、この第二のワンタイムＩＤと上記第二の乱数とを上記第一装置に対して送信するステップと、上記第一装置が、上記第一の乱数および上記第一の共有鍵に基づいて上記第二のワンタイムＩＤを演算により求め、この演算結果と上記第二装置から受信した上記第二のワンタイムＩＤとの比較により、上記第二装置の正当性を判定するステップと、上記第一装置が、上記第一の乱数および上記第二の乱数に基づいて第二の共有鍵を生成するとともに、この第二の共有鍵、上記第一の乱数および上記第二の乱数を引数とする一方向関数の関数値を第三のワンタイムＩＤ（$SIGNAL_{c2}$）として求め、この第三のワンタイムＩＤを上記第二装置に対して送信するステップと、上記第二装置が、上記第一の乱数および上記第二の乱数に基づいて上記第二の共有鍵を生成するとともに、この第二の共有鍵、上記第一の乱数および上記第二の乱数に基づいて上記第三のワンタイムＩＤを演算により求め、この演算結果と上記第一装置から受信した上記第三のワンタイムＩＤとの比較により、上記第一装置の正当性を判定するステップとを有することを特徴とするものである。

【００２８】

請求項１２に記載の発明は、請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いて、第一装置と第二装置間における認証（相互認証）を行う認証方法であって、上記第一装置が、第一の乱数を生成するとともに、上記第二装置との間で予め共有化された共有鍵を引数とする一方向関数の関数値を第一のワンタイムＩＤ（$SIGNAL_{c1}$）として求め、

この第一のワンタイムＩＤと上記第一の乱数を上記第二装置に対して送信するステップと、上記第二装置が、第二の乱数を生成するとともに、上記第一の乱数と上記共有鍵とを引数とする一方向関数の関数値を第二のワンタイムＩＤ（$SIGNAL_{s1}$）として求め、この第二のワンタイムＩＤと上記第二の乱数を上記第一装置に対して送信するステップと、上記第一装置が、上記第一の乱数および上記共有鍵に基づいて上記第二のワンタイムＩＤを演算により求め、この演算結果と上記第二装置から受信した上記第二のワンタイムＩＤとの比較により、上記第二装置の正当性を判定するステップと、上記第一装置が、上記第一の乱数、上記第二の乱数および上記共有鍵を引数とする一方向関数の関数値を第三のワンタイムＩＤ（$SIGNAL_{c2}$）として求め、この第三のワンタイムＩＤを上記第二装置に対して送信するステップと、上記第二装置が、上記第一の乱数、上記第二の乱数および上記共有鍵に基づいて上記第三のワンタイムＩＤを演算により求め、この演算結果と上記第一装置から受信した上記第三のワンタイムＩＤとの比較により、上記第一装置の正当性を判定するステップとを有することを特徴とするものである。

【００２９】

請求項１３に記載の発明は、請求項１１または請求項１２に記載の認証方法において、上記第一の乱数と上記第二の乱数を、上記第一装置と上記第二装置との間で予め共有化された共有鍵で暗号化した状態で、送信するようにしたことを特徴とするものである。

【００３０】

請求項１４に記載の発明は、請求項１１～請求項１３の何れかに記載の認証方法において、上記第二装置が上記第二のワンタイムＩＤと上記第二の乱数とを上記第一装置に対して送信するステップにおいて、上記第二装置は、上記第一装置との間で予め共有化された乱数を初期乱数として、この初期乱数と上記第一の乱数を引数とする所定の演算を行い、この演算結果を上記第一装置に対して送信する一方、上記第一装置は、上記第二装置の正当性の判定材料として、上記第二装置から受信した上記演算結果を、上記第二のワンタイムＩＤとともに用いることを特徴とするものである。

【００３１】

請求項１５に記載の発明は、請求項１１～請求項１４の何れかに記載の認証方法において、上記第一装置が上記第三のワンタイムＩＤを上記第二装置に対して送信するステップにおいて、上記第一装置は、上記第一の乱数と上記第二の乱数を引数とする所定の演算を行い、この演算結果を上記第二装置に対して送信する一方、上記第二装置は、上記第一装置の正当性の判定材料として、上記第一装置から受信した上記演算結果を、上記第三のワンタイムＩＤとともに用いることを特徴とするものである。

【００３２】

請求項１６に記載の発明は、請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いて、第一装置と第二装置間における認証を行う認証方法であって、上記第一装置が、第一の乱数を生成するとともに、上記第二装置との間で予め共有化された共有鍵、第一の記憶乱数および第二の記憶乱数を引数とする一方向関数の関数値を第一のワンタイムＩＤ（ＳＩＧＮＡＬ$_{ci}$）として求め、当該第一装置に予め設定されたＩＤ、上記第二装置に予め設定されたＩＤおよび上記第一の乱数を上記共有鍵で暗号化した第一の暗号化データと、上記第一のワンタイムＩＤとを上記第二装置に対して送信するステップと、上記第二装置が、上記第一のワンタイムＩＤを演算により求め、この演算結果と、上記第一装置から受信した上記第一のワンタイムＩＤとの照合により、上記第一装置を識別するステップと、上記第二装置が、上記第一装置を識別できた場合に、上記共有鍵を用いて上記第一の暗号化データを復号し、この復号したデータに含まれる、上記第一装置に予め設定されたＩＤおよび当該第二装置に予め設定されたＩＤに基づいて、上記第一装置の正当性を判定するステップと、上記第二装置が、上記第一装置を正当であると判定した場合に、第二の乱数を生成するとともに、上記第一の乱数、上記第二の記憶乱数および上記共有鍵を引数とする一方向関数の関数値を第二のワンタイムＩＤ（ＳＩＧＮＡＬ$_{si}$）として求め、上記第一装置に予め設定されたＩＤ、当該第二装置に予め設定されたＩＤおよび上記第二の乱数を上記共有鍵で暗号化した第二の暗号化データと、上記第二のワンタイムＩＤとを上記第一装置に対して送信するステップと、上記第二装置

が、上記第一の記憶乱数を上記第一の乱数に、上記第二の記憶乱数を上記第二の乱数にそれぞれ置換するステップと、上記第一装置が、上記第二のワンタイムＩＤを演算により求め、この演算結果と、上記第二装置から受信した上記第二のワンタイムＩＤとの照合により、上記第二装置を識別するステップと、上記第一装置が、上記第二装置を識別できた場合に、上記共有鍵を用いて上記第二の暗号化データを復号し、この復号したデータに含まれる、上記第二装置に予め設定されたＩＤおよび当該第一装置に予め設定されたＩＤに基づいて、上記第二装置の正当性を判定するステップと、上記第一装置が、上記第一の記憶乱数を上記第一の乱数に、上記第二の記憶乱数を上記第二の乱数にそれぞれ置換するステップとを有することを特徴とするものである。

【００３３】

請求項１７に記載の発明は、請求項１６に記載の認証方法において、上記第一の記憶乱数を上記第一の乱数に、上記第二の記憶乱数を上記第二の乱数にそれぞれ置換した後に、これら第一の記憶乱数および第二の記憶乱数に基づいて上記共有鍵を生成することにより、当該共有鍵を変化させるようにしたことを特徴とするものである。

【００３４】

請求項１８に記載の発明は、請求項１または請求項４に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤ（$SIGNAL_n$）を用いてクライアントとの間で認証を行うサーバであって、上記クライアントに予め設定されたクライアントＩＤを少なくとも引数とする一方向関数Ｆｃの関数値と、上記クライアントに予め記憶されたDiffie-Hellman公開値の一方と、上記ワンタイムＩＤとを上記クライアントから受信する受信手段と、上記一方向関数の関数値Ｆｃおよび上記ワンタイムＩＤを演算により求め、この演算結果と、上記クライアントから受信した上記ワンタイムＩＤおよび上記一方向関数Ｆｃの関数値との比較により、上記クライアントの正当性を判定する判定手段と、上記判定手段が上記クライアントを正当であると判定した場合に、当該サーバに予め設定されたサーバＩＤを少なくとも引数とする一方向関数Ｆｓの関数値と、当該サーバに予め記憶されたDiffie-Hellman公開値の他方とを上記クライアントに対して送信する送信

手段とを備えることを特徴とするものである。

【００３５】

請求項１９に記載の発明は、請求項１または請求項４に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤ（ＳＩＧＮＡＬ$_n$）を用いてサーバとの間で認証を行うクライアントであって、上記サーバとの間で予め共有化された可変共有鍵を用いて上記ワンタイムＩＤを生成するとともに、当該クライアントに予め設定されたクライアントＩＤを少なくとも引数とする一方向関数Ｆｃの関数値を演算により求め、これらワンタイムＩＤおよび一方向関数Ｆｃの関数値と、当該クライアントに予め記憶されたDiffie-Hellman公開値の一方とを上記サーバに対して送信する送信手段と、上記サーバに予め設定されたサーバＩＤを少なくとも引数とする一方向関数Ｆｓの関数値と、上記サーバに予め記憶されたDiffie-Hellman公開値の他方とを上記サーバから受信する受信手段と、上記一方向関数Ｆｓの関数値を演算により求め、この演算結果と、上記サーバから受信した上記一方向関数Ｆｓの関数値との比較により、上記サーバの正当性を判定する判定手段とを備えることを特徴とするものである。

【００３６】

請求項２０に記載の本発明に係る認証システムは、請求項１８に記載のサーバと、請求項１９に記載のクライアントとを備えてなることを特徴とするものである。

【００３７】

請求項２１に記載の発明は、請求項１または請求項４に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤ（ＳＩＧＮＡＬ$_n$）に基づいてクライアントとの間で認証を行うサーバに実行させるプログラムであって、上記クライアントに予め設定されたクライアントＩＤを少なくとも引数とする一方向関数Ｆｃの関数値と、上記クライアントに予め記憶されたDiffie-Hellman公開値の一方と、上記ワンタイムＩＤとを上記クライアントから受信する処理と、上記一方向関数の関数値Ｆｃおよび上記ワンタイムＩＤを演算により求め、この演算結果と、上記クライアントから受信した上記ワンタイムＩＤおよび上記一方向関数Ｆｃの関数値との比較により、上記クライアントの正当性を判定する処理と、上記

クライアントが正当であると判定された場合に、上記サーバに予め設定されたサーバＩＤを少なくとも引数とする一方向関数Ｆｓの関数値と、上記サーバに予め記憶されたDiffie-Hellman公開値の他方とを上記クライアントに対して送信する処理とを上記サーバに実行させることを特徴とするものである。

【００３８】

請求項２２に記載の発明は、請求項１または請求項４に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤ（$SIGNAL_n$）に基づいてサーバとの間で認証を行うクライアントに実行させるプログラムであって、上記サーバとの間で予め共有化された可変共有鍵を用いて上記ワンタイムＩＤを生成するとともに、上記クライアントに予め設定されたクライアントＩＤを少なくとも引数とする一方向関数Ｆｃの関数値を演算により求め、これらワンタイムＩＤおよび一方向関数Ｆｃの関数値と、上記クライアントに予め記憶されたDiffie-Hellman公開値の一方とを上記サーバに対して送信する処理と、上記サーバに予め設定されたサーバＩＤを少なくとも引数とする一方向関数Ｆｓの関数値と、上記サーバに予め記憶されたDiffie-Hellman公開値の他方とを上記サーバから受信する処理と、上記一方向関数Ｆｓの関数値を演算により求め、この演算結果と、上記サーバから受信した上記一方向関数Ｆｓの関数値との比較により、上記サーバの正当性を判定する処理とを上記クライアントに実行させることを特徴とするものである。

【００３９】

請求項２３に記載の発明は、請求項２または請求項５に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてクライアントとの間で認証を行うサーバであって、上記クライアントとの間で予め共有化された第一の可変共有鍵と上記クライアントの通信順序に関する情報とを引数とする一方向関数の関数値を第一のワンタイムＩＤ（$SIGNAL_{n,j}$）として、この第一のワンタイムＩＤ、上記クライアントに予め設定されたクライアントＩＤ、当該サーバに予め設定されたサーバＩＤ、上記クライアントに予め記憶されたDiffie-Hellman公開値の一方を上記第一の可変共有鍵で暗号化した暗号化データと、上記第一のワンタイムＩＤとを上記クライアントから受信する受信手段と、上記第一のワン

タイムＩＤを演算により求め、この演算結果と、上記クライアントから受信した上記第一のワンタイムＩＤとの照合により、上記クライアントを識別し、上記クライアントを識別できた場合に、上記第一の可変共有鍵を用いて上記暗号化データを復号し、この復号したデータに含まれる、上記クライアントＩＤ、上記サーバＩＤおよび上記第一のワンタイムＩＤに基づいて、上記クライアントの正当性を判定する判定手段と、上記判定手段が上記クライアントを正当であると判定した場合に、上記第一の可変共有鍵と当該サーバの通信順序に関する情報とを引数とする一方向関数の関数値を第二のワンタイムＩＤ（$SIGNAL'_{n,1}$）として生成するとともに、上記クライアントから受信したDiffie-Hellman公開値の一方と当該サーバに予め記憶されたDiffie-Hellman公開値の他方とからDiffie-Hellman共通鍵を第二の可変共有鍵として生成し、この第二の可変共有鍵、上記クライアントＩＤ、上記サーバＩＤおよび上記第二のワンタイムＩＤを引数とする一方向関数ｈの関数値と、上記Diffie-Hellman公開値の他方と、上記第二のワンタイムＩＤとを上記クライアントに対して送信する送信手段とを備えることを特徴とするものである。

【００４０】

請求項２４に記載の発明は、請求項２または請求項５に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてサーバとの間で認証を行うクライアントであって、上記サーバとの間で予め共有化された第一の可変共有鍵と当該クライアントの通信順序に関する情報とを引数とする一方向関数の関数値を第一のワンタイムＩＤ（$SIGNAL_{n,j}$）として生成するとともに、上記第一の可変共有鍵を用いて、当該クライアントに予め設定されたクライアントＩＤ、上記サーバに予め設定されたサーバＩＤ、当該クライアントに予め記憶されたDiffie-Hellman公開値の一方および上記第一のワンタイムＩＤを暗号化し、この暗号化データと上記第一のワンタイムＩＤとを上記サーバに対して送信する送信手段と、上記第一の可変共有鍵と上記サーバの通信順序に関する情報とを引数とする一方向関数の関数値を第二のワンタイムＩＤ（$SIGNAL'_{n,1}$）とし、Diffie-Hellman共通鍵を第二の可変共有鍵として、上記第二のワンタイムＩＤ、上記第二の可変共有鍵、上記クライアントＩＤおよび上記サーバＩＤを引数とす

る一方向関数ｈの関数値と、上記サーバに予め記憶されたDiffie-Hellman公開値の他方と、上記第二のワンタイムＩＤとを上記サーバから受信する受信手段と、上記第二のワンタイムＩＤを演算により求め、この演算結果と、上記サーバから受信した上記第二のワンタイムＩＤとの照合により、上記サーバを識別し、上記サーバを識別した場合に、上記サーバから受信した上記Diffie-Hellman公開値の他方と当該クライアントに予め記憶された上記Diffie-Hellman公開値の一方とからDiffie-Hellman共通鍵を上記第二の可変共有鍵として生成するとともに、この第二の可変共有鍵を用いて上記一方向関数ｈの関数値を演算により求め、この演算結果と、上記サーバから受信した一方向関数ｈの関数値との照合により、上記サーバの正当性を判定する判定手段とを備えることを特徴とするものである。

【００４１】

請求項２５に記載の本発明に係る認証システムは、請求項２３に記載のサーバと、請求項２４に記載のクライアントとを備えてなることを特徴とするものである。

【００４２】

請求項２６に記載の発明は、請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてクライアントとの間で相互に認証を行うサーバであって、上記クライアントとの間で予め共有化された第一の共有鍵を引数とする一方向関数の関数値を第一のワンタイムＩＤ（$SIGNAL_{c1}$）として、この第一のワンタイムＩＤと、上記クライアントで生成された第一の乱数とを上記クライアントから受信する第一受信手段と、第二の乱数を生成するとともに、上記第一の乱数と上記第一の共有鍵とを引数とする一方向関数の関数値を第二のワンタイムＩＤ（$SIGNAL_{s1}$）として求め、この第二のワンタイムＩＤと上記第二の乱数とを上記クライアントに対して送信する送信手段と、上記第一の乱数、上記第二の乱数および第二の共有鍵を引数とする一方向関数の関数値を第三のワンタイムＩＤ（$SIGNAL_{c2}$）として、この第三のワンタイムＩＤを上記クライアントから受信する第二受信手段と、上記第一の乱数および上記第二の乱数に基づいて上記第二の共有鍵を生成するとともに、この第二の共有鍵、上記第一の乱数および上記第二の乱数に基づいて上記第三のワンタイム

ＩＤを演算により求め、この演算結果と上記クライアントから受信した上記第三のワンタイムＩＤとの比較により、上記クライアントの正当性を判定する判定手段とを備えることを特徴とするものである。

【００４３】

請求項２７に記載の発明は、請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてサーバとの間で相互に認証を行うクライアントであって、第一の乱数を生成するとともに、上記サーバとの間で予め共有化された第一の共有鍵を引数とする一方向関数の関数値を第一のワンタイムＩＤ（$SIGNAL_{c1}$）として求め、この第一のワンタイムＩＤと上記第一の乱数とを上記サーバに対して送信する第一送信手段と、上記第一の乱数と上記第一の共有鍵とを引数とする一方向関数の関数値を第二のワンタイムＩＤ（$SIGNAL_{s1}$）として、この第二のワンタイムＩＤと、上記サーバで生成された第二の乱数とを上記サーバから受信する受信手段と、上記第一の乱数および上記第一の共有鍵に基づいて上記第二のワンタイムＩＤを演算により求め、この演算結果と上記サーバから受信した上記第二のワンタイムＩＤとの比較により、上記サーバの正当性を判定する判定手段と、上記判定手段により上記サーバが正当であると判定された場合に、上記第一の乱数および上記第二の乱数に基づいて第二の共有鍵を生成するとともに、この第二の共有鍵、上記第一の乱数および上記第二の乱数を引数とする一方向関数の関数値を第三のワンタイムＩＤ（$SIGNAL_{c2}$）として求め、この第三のワンタイムＩＤを上記サーバに対して送信する第二送信手段とを備えることを特徴とするものである。

【００４４】

請求項２８に記載の本発明に係る認証システムは、請求項２６に記載のサーバと、請求項２７に記載のクライアントとを備えてなることを特徴とするものである。

【００４５】

請求項２９に記載の発明は、請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてクライアントとの間で相互に認証を行うサーバであって、上記クライアントとの間で予め共有化された共有

鍵を引数とする一方向関数の関数値を第一のワンタイムＩＤ（$SIGNAL_{c1}$）として、この第一のワンタイムＩＤと、上記クライアントで生成された第一の乱数とを上記クライアントから受信する第一受信手段と、第二の乱数を生成するとともに、上記第一の乱数と上記共有鍵とを引数とする一方向関数の関数値を第二のワンタイムＩＤ（$SIGNAL_{s1}$）として求め、この第二のワンタイムＩＤと上記第二の乱数を上記クライアントに対して送信する送信手段と、上記共有鍵、上記第一の乱数および上記第二の乱数を引数とする一方向関数の関数値を第三のワンタイムＩＤ（$SIGNAL_{c2}$）として、この第三のワンタイムＩＤを上記クライアントから受信する第二受信手段と、上記第一の乱数、上記第二の乱数および上記共有鍵に基づいて上記第三のワンタイムＩＤを演算により求め、この演算結果と上記クライアントから受信した上記第三のワンタイムＩＤとの比較により、上記クライアントの正当性を判定する判定手段とを備えることを特徴とするものである。

【００４６】

請求項３０に記載の発明は、請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてサーバとの間で相互に認証を行うクライアントであって、第一の乱数を生成するとともに、上記サーバとの間で予め共有化された共有鍵を引数とする一方向関数の関数値を第一のワンタイムＩＤ（$SIGNAL_{c1}$）として求め、この第一のワンタイムＩＤと上記第一の乱数を上記サーバに対して送信する第一送信手段と、上記第一の乱数と上記共有鍵とを引数とする一方向関数の関数値を第二のワンタイムＩＤ（$SIGNAL_{s1}$）として、この第二のワンタイムＩＤと、上記サーバで生成された第二の乱数とを上記サーバから受信する受信手段と、上記第一の乱数および上記共有鍵に基づいて上記第二のワンタイムＩＤを演算により求め、この演算結果と上記サーバから受信した上記第二のワンタイムＩＤとの比較により、上記サーバの正当性を判定する判定手段と、上記判定手段により上記サーバが正当であると判定された場合に、上記第一の乱数、上記第二の乱数および上記共有鍵を引数とする一方向関数の関数値を第三のワンタイムＩＤ（$SIGNAL_{c2}$）として求め、この第三のワンタイムＩＤを上記サーバに対して送信する第二送信手段とを備えることを特

徴とするものである。

【００４７】

請求項３１に記載の本発明に係る認証システムは、請求項２９に記載のサーバと、請求項３０に記載のクライアントとを備えてなることを特徴とするものである。

【００４８】

請求項３２に記載の発明は、請求項３または請求項６に記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いてクライアントとの間で相互に認証を行うサーバであって、上記クライアントとの間で予め共有化された共有鍵、第一の記憶乱数および第二の記憶乱数を引数とする一方向関数の関数値を第一のワンタイムＩＤ（$SIGNAL_{ci}$）として、この第一のワンタイムＩＤを上記クライアントから受信するとともに、上記クライアントで生成された第一の乱数、上記クライアントに予め設定されたクライアントＩＤ、当該サーバに予め設定されたサーバＩＤを上記共有鍵で暗号化した第一の暗号化データを上記クライアントから受信する受信手段と、上記第一のワンタイムＩＤを演算により求め、この演算結果と、上記クライアントから受信した上記第一のワンタイムＩＤとの照合により、上記クライアントを識別し、上記クライアントを識別できた場合に、上記共有鍵を用いて上記第一の暗号化データを復号し、この復号したデータに含まれる上記クライアントＩＤおよび上記サーバＩＤに基づいて、上記クライアントの正当性を判定する判定手段と、上記判定手段が上記クライアントを正当であると判定した場合に、第二の乱数を生成するとともに、上記第一の乱数、上記第二の記憶乱数および上記共有鍵を引数とする一方向関数の関数値を第二のワンタイムＩＤ（$SIGNAL_{si}$）として求め、上記クライアントＩＤ、上記サーバＩＤおよび上記第二の乱数を上記共有鍵で暗号化した第二の暗号化データと、上記第二のワンタイムＩＤとを上記クライアントに対して送信する送信手段と、上記第一の記憶乱数を上記第一の乱数に、上記第二の記憶乱数を上記第二の乱数にそれぞれ置換する置換手段とを備えることを特徴とするものである。

【００４９】

請求項３３に記載の発明は、請求項３または請求項６に記載のワンタイムＩＤ

の生成方法により生成されたワンタイムＩＤを用いてサーバとの間で相互に認証を行うクライアントであって、第一の乱数を生成するとともに、上記サーバとの間で予め共有化された共有鍵、第一の記憶乱数および第二の記憶乱数を引数とする一方向関数の関数値を第一のワンタイムＩＤ（$SIGNAL_{ci}$）として求め、当該クライアントに予め設定されたクライアントＩＤ、上記サーバに予め設定されたサーバＩＤおよび上記第一の乱数を上記共有鍵で暗号化した第一の暗号化データと、上記第一のワンタイムＩＤとを上記サーバに対して送信する送信手段と、上記第一の乱数、上記第二の記憶乱数および上記共有鍵を引数とする一方向関数の関数値を第二のワンタイムＩＤ（$SIGNAL_{si}$）として、この第二のワンタイムＩＤを上記サーバから受信するとともに、上記サーバで生成された第二の乱数、上記クライアントＩＤおよび上記サーバＩＤを上記共有鍵で暗号化した第二の暗号化データを上記サーバから受信する受信手段と、上記第二のワンタイムＩＤを演算により求め、この演算結果と、上記サーバから受信した上記第二のワンタイムＩＤとの照合により、上記サーバを識別し、上記サーバを識別できた場合に、上記共有鍵を用いて上記第二の暗号化データを復号し、この復号したデータに含まれる上記サーバＩＤおよび上記クライアントＩＤに基づいて、上記サーバの正当性を判定する判定手段と、上記第一の記憶乱数を上記第一の乱数に、上記第二の記憶乱数を上記第二の乱数にそれぞれ置換する置換手段とを備えることを特徴とするものである。

【００５０】

請求項３４に記載の本発明に係る認証システムは、請求項３２に記載のサーバと、請求項３３に記載のクライアントとを備えてなることを特徴とするものである。

【００５１】

請求項３５に記載の発明は、請求項３４に記載の認証システムにおいて、上記サーバおよび上記クライアントは、上記第一の記憶乱数を上記第一の乱数に、上記第二の記憶乱数を上記第二の乱数にそれぞれ置換した後で、これら第一の記憶乱数および第二の記憶乱数に基づいて上記共有鍵を生成することにより、当該共有鍵を変化させるようになっていることを特徴とするものである。

【００５２】

請求項１または請求項４に記載の発明によれば、可変共有鍵を引数とする一方向関数の関数値を求め、この関数値からワンタイムＩＤを生成するようにしたため、例えば、可変共有鍵が第三者に漏れたとしても、所定の通信単位毎に可変共有鍵が変化することとなるので、漏れた可変共有鍵を用いて生成されたワンタイムＩＤ以外のワンタイムＩＤを予測することはできない。すなわち、盗聴が困難で安全性に優れたワンタイムＩＤを生成することが可能になり、ワンタイムＩＤの将来にわたる安全性（ＰＦＳ）を実現することが可能になる。

【００５３】

請求項２または請求項５に記載の発明によれば、可変共有鍵と通信順序または回数に関する情報とを引数とする一方向関数の関数値を求め、この関数値からワンタイムＩＤを生成するようにしたため、例えば、可変共有鍵が第三者に漏れたとしても、所定の通信単位毎に可変共有鍵が変化するとともに、各通信毎に通信順序または回数に関する情報も変化することとなるので、漏れた可変共有鍵を用いて生成されたワンタイムＩＤ以外のワンタイムＩＤを予測することは事実上不可能となり、また漏れた可変共有鍵を用いて生成されたワンタイムＩＤの予測自体も非常に困難なものとなる。すなわち、盗聴が困難で安全性に優れたワンタイムＩＤを生成することが可能になり、ワンタイムＩＤの将来にわたる安全性（ＰＦＳ）を実現することが可能になる。

【００５４】

請求項３または請求項６に記載の発明によれば、所定の通信単位内で生成された乱数と所定の共有鍵とを引数とする一方向関数の関数値を求め、この関数値からワンタイムＩＤを生成するようにしたため、例えば、共有鍵が第三者に漏れたとしても、乱数によって一方向関数の関数値が所定の通信単位毎に変化することとなるので、所定の通信単位内で生成される乱数がわからない限り、ワンタイムＩＤを予測することはできない。すなわち、盗聴が困難で安全性に優れたワンタイムＩＤを生成することが可能になり、ワンタイムＩＤの将来にわたる安全性（ＰＦＳ）を実現することが可能になる。

【００５５】

請求項７～請求項３５の何れかに記載の発明によれば、請求項１～請求項６の何れかに記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いて、装置間（クライアント・サーバ間）における認証を行うようにしたので、第三者（攻撃者）が送信者・受信者を特定できなくなる一方で、正当な送信者・受信者であればワンタイムＩＤを識別情報として把握することができる。

したがって、ＤｏＳ攻撃やなりすまし等に対する耐性を強化することができ、オープンなネットワーク環境下においても、ＩＤ情報の保護を図り、通信の安全性を向上させることができる。また、リモートアクセスが可能になり、利便性の向上を図ることができる。

【００５６】

請求項８に記載の発明によれば、第一装置の正当性を判定するのに用いる一方向関数Ｆｃとして、所定の共有鍵、Diffie-Hellman公開値の一方、第一装置に予め設定されたＩＤ、ワンタイムＩＤを引数とする疑似乱数関数を用いるとともに、第二装置の正当性を判定するのに用いる一方向関数Ｆｓとして、所定の共有鍵、Diffie-Hellman公開値の一方、Diffie-Hellman公開値の他方、第二装置に予め設定されたＩＤ、ワンタイムＩＤを引数とする疑似乱数関数を用いるようにしたので、従来の鍵交換・認証方式では３回必要であった通信回数を２回に低減することが可能になり、迅速かつ安全な認証および鍵交換を実現することが可能になる。

【００５７】

【発明の実施の形態】

［第１の実施形態］

図１は、本発明に係る認証システムの一実施形態を示す概略構成図である。この認証システムは、公衆回線網やインターネット等のネットワーク４０を介して相互に接続されたサーバ（第二装置）１０とクライアント（第一装置）２０とにより概略構成されている。この実施形態では、種々のサービスを提供する複数のサーバＡ、Ｂ、Ｃ、…がサーバ１０に接続され、当該サーバ１０が、サーバＡ、Ｂ、Ｃ、…へのアクセスの可否を決定する認証サーバとして機能するようになっている。

【００５８】

サーバ１０は、図２に示すように、ＣＰＵ１１、ＲＡＭ１２、記憶装置１３、入力装置１４、表示装置１５および通信装置１６等により構成され、各部はバス１７により接続されている。

【００５９】

ＣＰＵ（Central Processing Unit）１１は、記憶装置１３の記憶領域に格納されている各種処理プログラム、入力装置１４や通信装置１６から入力される各種指示、あるいは指示に対応する各種データ等をＲＡＭ１２に格納し、それら入力指示および各種データに応じてＲＡＭ１２に格納した各種処理プログラムに従って各種処理を実行し、その処理結果をＲＡＭ１２に一時的に記憶するとともに、表示装置１５等に出力する。

【００６０】

このＣＰＵ１１は、当該サーバ１０における受信手段および判定手段を構成しており、クライアントＩＤを引数とする一方向関数（一方向関数Ｆｃ）の関数値であるＨＡＳＨｃ、ワンタイムＩＤ（ＳＩＧＮＡＬ）、ＤＨ公開値g^x（Diffie-Hellman公開値の一方）をクライアント２０から受信した場合（すなわち、クライアント２０からアクセスの要求を受けた場合）に、クライアント２０から受信した受信データと記憶装置１３に記憶されている記憶データを用いてワンタイムＩＤおよびＨＡＳＨｃを演算により求め、この演算結果と、クライアント２０から受信したワンタイムＩＤおよびＨＡＳＨｃとの比較により、クライアント２０の正当性を判定する処理を実行する。

【００６１】

また、ＣＰＵ１１は、当該サーバ１０における送信手段を構成しており、クライアント２０が正当であると判定される場合に、上記受信データおよび上記記憶データを用いて、サーバＩＤを引数とする一方向関数（一方向関数Ｆｓ）の関数値であるＨＡＳＨｓを演算により求め、このＨＡＳＨｓと、記憶装置１３に記憶されているＤＨ公開値g^y（Diffie-Hellman公開値の他方）とをクライアント２０に対して送信する処理を実行する。

【００６２】

なお、上記ワンタイムＩＤ（ＳＩＧＮＡＬ）は、サーバ・クライアント間における認証において一回限り使用可能な識別情報であり、このワンタイムＩＤを生成する場合には、所定の通信単位毎に変化する暗号化鍵Ｋ（可変共有鍵）を記憶装置１３から読み込んで、この暗号化鍵Ｋを引数とするハッシュ関数（一方向関数）の関数値を求め、この関数値から上記ワンタイムＩＤを生成する。

【００６３】

ＲＡＭ（Random Access Memory）１２は、クライアント２０等との間で送受信されるデータなど、認証に関する各種データを一時記憶する記憶領域や、ＣＰＵ１１の作業領域などを備えている。

【００６４】

記憶装置１３は、プログラムやデータ等が記憶される記憶媒体１３ａを有し、この記憶媒体１３ａは磁気的、光学的記録媒体、若しくは半導体メモリで構成されている。この記憶媒体１３ａは記憶装置１３に固定的に設けたもの、若しくは着脱自在に装着するものであり、ＣＰＵ１１により実行される各種処理プログラムや制御データ等を記憶する記憶領域、認証に関する各種データ（例えば、クライアント２０やＩＤ発行管理サーバ３０（後述）から取得したデータ、認証の処理過程で生成されたデータなど）を格納する記憶領域などを備えている。なお、この記憶媒体１３ａに記憶するプログラムやデータなどは、その一部若しくは全部を他のサーバ等からネットワーク４０を介して受信して記憶する構成とすることも可能である。この記憶媒体１３ａには、サーバＩＤ、ＤＨ公開値g^y、クライアント２０との間で共有化された乱数Ｒなどが、認証処理を開始する前段階で予め格納された状態となっている。

【００６５】

入力装置１４は、キーボードやポインティングデバイス等により構成され、入力指示信号をＣＰＵ１１に対して出力する。

表示装置１５は、ＣＲＴ（Cathode Ray Tube）やＬＣＤ（Liquid Crystal Display）等により構成され、ＣＰＵ１１から入力される表示データを表示する。

通信装置１６は、モデムやルータ、ブリッジ等により構成され、ネットワーク４０を介してクライアント２０等より受信したデータをＣＰＵ１１に出力すると

ともに、ＣＰＵ１１より受信したデータをネットワーク４０を介してクライアント２０等に対して出力する。

【００６６】

一方、クライアント２０は、図３に示すように、ＣＰＵ２１、ＲＡＭ２２、記憶装置２３、入力装置２４、表示装置２５および通信装置２６等により構成され、各部はバス２７により接続されている。具体的に、クライアント２０としては、例えば、パーソナルコンピュータや、ＰＤＡ（Personal Digital Assistance）等の携帯情報端末、インターネット接続サービスを利用可能な携帯電話などが挙げられる。なお、このクライアント２０の各構成要素は、前述したサーバ１０の各構成要素とほぼ同様であるので、相違点のみを以下に説明する。

【００６７】

すなわち、クライアント２０のＣＰＵ２１は、当該クライアント２０における送信手段を構成しており、入力装置２４からの指示入力等に基づいて、ワンタイムＩＤ（ＳＩＧＮＡＬ）を生成するとともに、クライアントＩＤを引数とする一方向関数（一方向関数Ｆｃ）の関数値であるＨＡＳＨｃを求め、これらワンタイムＩＤおよびＨＡＳＨｃと、記憶装置２３に予め記憶されたＤＨ公開値g^x（Diffie-Hellman公開値の一方）とをサーバ１０に対して送信する処理を実行する。

【００６８】

また、ＣＰＵ２１は、当該クライアント２０における受信手段および判定手段を構成しており、サーバＩＤを引数とする一方向関数（一方向関数Ｆｓ）の関数値であるＨＡＳＨｓと、ＤＨ公開値g^y（Diffie-Hellman公開値の他方）とをサーバ１０から受信した場合（すなわち、サーバ１０によってクライアント２０が正当であると判定された場合）に、サーバ１０から受信した受信データと記憶装置２３に記憶されている記憶データを用いてＨＡＳＨｓを演算により求め、この演算結果と、サーバ１０から受信したＨＡＳＨｓとの比較により、サーバ１０の正当性を判定する処理を実行する。

【００６９】

記憶装置２３は、プログラムやデータ等が記憶される記憶媒体２３ａを有し、この記憶媒体２３ａは、上記ＣＰＵ２１により実行される各種処理プログラムや

制御データ等を記憶する記憶領域、認証に関する各種データ（例えば、サーバ１０やＩＤ発行管理サーバ３０（後述）から取得したデータ、認証の処理過程で生成されたデータなど）を格納する記憶領域などを備えている。この記憶媒体２３ａには、クライアントＩＤ、ＤＨ公開値g^x、サーバ１０との間で共有化された乱数Ｒなどが、認証処理を開始する前段階で予め格納された状態となっている。

【００７０】

ＩＤ発行管理サーバ３０は、クライアント・サーバ間で共有化される秘密情報（例えば、ワンタイムＩＤの初期値を生成するのに用いられる乱数Ｒなど）や、クライアントＩＤ、サーバＩＤなどを発行・管理するためのサーバである。このＩＤ発行管理サーバ３０は、クライアント２０を利用するユーザのＩＤ（例えば、クレジットＮｏ、住基ネットＩＤ、社員Ｎｏ、学生Ｎｏ、特定会員Ｎｏなど）に上記秘密情報やパスワードなどを対応付けた状態で格納するデータベースを有している。また、ＩＤ発行管理サーバ３０は、一定の周期で上記データベース内の秘密情報を更新し、この更新した秘密情報をオンライン（例えば、電子メールなど）またはオフライン（例えば、郵送など）で、クライアント２０とサーバ１０の双方に配布するようになっている。なお、上記秘密情報の発行は、クライアント２０またはサーバ１０からの発行依頼に基づくものであってもよい。

【００７１】

次に、上記構成からなる認証システムによって行われる認証方法の第１の実施形態について、図４に基づいて説明する。この方法は、ＲＦＣ２４０９において規定されたＩＫＥの方式に、本発明に係るワンタイムＩＤ（ＳＩＧＮＡＬ）を適用したものである。

【００７２】

先ず、ステップＳ１では、ＩＫＥによるＳＡ生成に際してイニシエータとなるクライアント２０が、ワンタイムＩＤ（ＳＩＧＮＡＬ）を生成するとともに、ＨＡＳＨｃを演算により求め、これらワンタイムＩＤおよびＨＡＳＨｃと、記憶装置２３に記憶されたＤＨ公開値g^xとをＳＡの提案とともに、レスポンダとなるサーバ１０に対して送信する処理を実行する。

【００７３】

ここで、ワンタイムＩＤであるＳＩＧＮＡＬは、例えば、ハッシュ関数を用いて、次のように生成される。

【００７４】

【数２】

$$SIGNAL_1 = R$$
$$SIGNAL_2 = hash(K_1)$$
$$SIGNAL_3 = hash(K_2)$$
$$\cdots$$
$$SIGNAL_n = hash(K_{n-1})$$

【００７５】

上記ＳＩＧＮＡＬの定義式において、ｈａｓｈはハッシュ関数、ＲはＩＤ発行管理サーバ３０からサーバ１０とクライアント２０の双方に発行されて両者間で共有化された乱数、K_i はｉ番目のセッションで生成されたサーバ・クライアント共有の暗号化鍵（可変共有鍵）である。なお、上記セッションは、ＳＡを確立してから当該ＳＡが無効になるまでの通信単位を示している。

【００７６】

すなわち、上記ＳＩＧＮＡＬの定義式によれば、前回のセッションで生成された上記暗号化鍵Ｋを引数とするハッシュ関数の関数値を求め、この関数値を今回のセッションのＳＩＧＮＡＬとして用いるようにしている。また、最初のセッションでは、サーバ・クライアント間で予め共有化された乱数ＲをＳＩＧＮＡＬの初期値として用いるようにしている。また、上記暗号化鍵K_i は、例えば、次式（１）から求められる。

【００７７】

$$K_i = prf(\text{共有鍵}, g^{xy}, SIGNAL_i) \quad \cdots(1)$$

【００７８】

この式（１）において、g^{xy}はＤＨ共通鍵であり、共有鍵は、サーバ・クライアント間の任意の共有鍵である。

【００７９】

一方、ＨＡＳＨｃは、次式（２）に示すように、共有鍵、ＤＨ公開値g^x、Ｉ

Ｄｃ（クライアントＩＤ）およびＳＩＧＮＡＬを引数とする疑似乱数関数（鍵付きハッシュ関数）の関数値として求められる。

【００８０】

ＨＡＳＨｃ＝ｐｒｆ（共有鍵、g^x、ＩＤｃ、ＳＩＧＮＡＬ）　　‥（２）

【００８１】

次いで、ステップＳ２では、サーバ１０が、ＳＩＧＮＡＬとＨＡＳＨｃを演算により求め、これら演算結果と、クライアント２０から受信したＳＩＧＮＡＬおよびＨＡＳＨｃとの比較により、クライアント２０の正当性を判定する処理を実行する。

【００８２】

上記判定の結果、受信データと演算結果とが一致して、クライアント２０が正当であると判定される場合には、ＨＡＳＨｓを演算により求め、このＨＡＳＨｓと、記憶装置１３に記憶されているＤＨ公開値g^yとを、受諾したＳＡとともにクライアント２０に対して送信する処理を実行する（ステップＳ３）。一方、受信データと演算結果とが一致せず、クライアント２０が正当でないと判定される場合には、クライアント２０からのアクセスを拒否して、当該認証処理を終了する。

【００８３】

ここで、ＨＡＳＨｓは、次式（３）に示すように、共有鍵、ＤＨ公開値g^x、g^y、ＩＤｓ（サーバＩＤ）およびＳＩＧＮＡＬを引数とする疑似乱数関数（鍵付きハッシュ関数）の関数値として求められる。

【００８４】

ＨＡＳＨｓ＝ｐｒｆ（共有鍵、g^x、g^y、ＩＤｓ、ＳＩＧＮＡＬ）‥（３）

【００８５】

また、このステップＳ３においては、記憶装置１３に記憶されているＤＨ公開値g^yと、クライアント２０から受信したＤＨ公開値g^xとからＤＨ共通鍵g^{xy}を生成して、ＤＨ共通鍵g^{xy}を記憶装置１３に格納する処理も併せて行う。

【００８６】

次いで、ステップＳ４では、クライアント２０が、ＨＡＳＨｓを演算により求

め、この演算結果と、サーバ１０から受信したＨＡＳＨｓとの比較により、サーバ１０の正当性を判定する処理を実行する。

【００８７】

上記判定の結果、受信データと演算結果とが一致して、サーバ１０が正当であると判定される場合には、記憶装置２３に記憶されているＤＨ公開値g^xと、サーバ１０から受信したＤＨ公開値g^yとからＤＨ共通鍵g^{xy}を生成して記憶装置２３に格納した後、当該認証処理を終了して、次のデータ伝送処理に移行する。一方、受信データと演算結果とが一致せず、サーバ１０が正当でないと判定される場合には、サーバ１０へのアクセスを中止して、当該認証処理を終了する。

【００８８】

以上のように、この第１の実施形態によれば、セッション毎に変化する暗号化鍵Ｋ（可変共有鍵）を引数とするハッシュ関数の関数値をワンタイムＩＤ（ＳＩＧＮＡＬ）として用いるようにしたので、例えば、暗号化鍵Ｋが第三者に漏れたとしても、セッション毎に暗号化鍵Ｋが変化することとなるので、漏れた暗号化鍵Ｋを用いて生成されたワンタイムＩＤ以外のワンタイムＩＤを予測できなくなる。すなわち、盗聴が困難で安全性に優れたワンタイムＩＤを生成することが可能になり、ワンタイムＩＤの将来にわたる安全性（ＰＦＳ）を実現することが可能になる。

【００８９】

また、上記ワンタイムＩＤ（ＳＩＧＮＡＬ）を用いて、クライアント・サーバ間における認証を行うようにしたので、第三者が送信者・受信者を特定できなくなる一方で、正当な送信者・受信者であればワンタイムＩＤを識別情報として把握することができる。

したがって、ＤｏＳ攻撃やなりすまし等に対する耐性を強化することができ、オープンなネットワーク環境下においても、ＩＤ情報の保護を図り、通信の安全性を向上させることができる。また、リモートアクセスが可能になり、利便性の向上を図ることもできる。

【００９０】

また、この実施形態では、クライアント２０の正当性を判定するのに用いる一

方向関数Ｆｃとして、共有鍵、ＤＨ公開値g^x、ＩＤｃ（クライアントＩＤ）およびＳＩＧＮＡＬを引数とする疑似乱数関数を用いるとともに、サーバ１０の正当性を判定するのに用いる一方向関数Ｆｓとして、共有鍵、ＤＨ公開値g^x、g^y、ＩＤｓ（サーバＩＤ）およびＳＩＧＮＡＬを引数とする疑似乱数関数を用いるようにしたので、従来の鍵交換・認証方式では３回必要であった通信回数を２回に低減することが可能になり、迅速かつ安全な認証および鍵交換を実現することが可能になる。

【００９１】

[第２の実施形態]

前述した第１の実施形態では、前回のセッションで生成された暗号化鍵（可変共有鍵）を引数とするハッシュ関数の関数値を求め、この関数値を今回のセッションのワンタイムＩＤ（ＳＩＧＮＡＬ）として用いるようにしたが、この第２の実施形態では、前回のセッションで生成された共有鍵と、当該セッションにおける通信順序とを引数とするハッシュ関数の関数値を求め、この関数値を今回のセッションの各通信時におけるワンタイムＩＤとして用いるようにしている。この第２の実施形態特有の部分以外は、第１の実施形態におけると同様である。この第２の実施形態において、第１の実施形態と同一部分には同一符号を付し、その説明を省略する。

【００９２】

図５は本発明に係る認証方法の第２の実施形態を説明する図である。この第２の実施形態では、先ず、ステップＰ１において、クライアント２０が、$SIGNAL_{n,1}$（第一のワンタイムＩＤ）を生成するとともに、ＩＤｃ（クライアントＩＤ）、ＩＤｓ（サーバＩＤ）、ＤＨ公開値g^{xn}および$SIGNAL_{n,1}$を共有鍵K_{n-1}（第一の可変共有鍵）で暗号化し、この暗号化データと$SIGNAL_{n,1}$とをサーバ１０に対して送信する処理を実行する。

【００９３】

ここで、ＳＩＧＮＡＬは、ｉ番目のセッションにおけるクライアント２０のｊ番目の通信で利用するＳＩＧＮＡＬを$SIGNAL_{i,j}$、ｉ番目のセッションにおけるサーバ１０のｊ番目の通信で利用するＳＩＧＮＡＬを$SIGNAL'_{i,j}$

とした場合、次のように生成される。

【００９４】

【数３】

$$SIGNAL_{1,j} = hash(R, j) \qquad i = 1$$

$$SIGNAL_{i,j} = hash(K_{i-1}, j) \qquad i \geqq 2$$

$$SIGNAL'_{1,j} = hash'(R, j) \qquad i = 1$$

$$SIGNAL'_{i,j} = hash'(K_{i-1}, j) \qquad i \geqq 2$$

【００９５】

上記ＳＩＧＮＡＬの定義式において、ｈａｓｈとｈａｓｈ’は互いに異なるハッシュ関数、ＲはＩＤ発行管理サーバ３０からサーバ１０とクライアント２０の双方に発行されて両者間で共有化された乱数、K_i はｉ番目のセッションで共有したＤＨ共通鍵g^{xiyi}（共有鍵）である。

【００９６】

すなわち、上記ＳＩＧＮＡＬの定義式によれば、前回のセッションで生成された共有鍵K_{i-1}と今回のセッションにおける通信順序ｊとを引数とするハッシュ関数の関数値を求め、この関数値を今回のセッションのｊ番目の通信に用いるＳＩＧＮＡＬとしている。ただし、最初のセッション（ｉ＝１）では、サーバ・クライアント間で予め共有化された乱数Ｒと当該セッションにおける通信順序ｊとを引数とするハッシュ関数の関数値を求め、この関数値を最初のセッションのｊ番目の通信に用いるＳＩＧＮＡＬとしている。

【００９７】

次いで、ステップＰ２では、サーバ１０が、$SIGNAL_{n,1}$ を演算により求め、この演算結果と、クライアント２０から受信した$SIGNAL_{n,1}$との照合により、クライアント２０を識別し、識別できない場合には、通信を拒否する。識別できる場合には、共有鍵K_{i-1}を用いて暗号化データを復号し、この復号したデータに含まれる、ＩＤｃ、ＩＤｓおよび$SIGNAL_{n,1}$に基づいて、クライアント２０の正当性を判定する処理を実行する。

【００９８】

上記判定の結果、受信データと、サーバ１０に予め格納された記憶データとが

一致して、クライアント２０が正当であると判定される場合には、前述したＳＩＧＮＡＬの定義式に従ってＳＩＧＮＡＬ’$_{n,1}$（第二のワンタイムＩＤ）を生成するとともに、クライアント２０から受信したＤＨ公開値g^{xn}と当該サーバ１０に予め記憶されたＤＨ公開値g^{yn}とからＤＨ共通鍵g^{xnyn}を共有鍵K_n（第二の可変共有鍵）として生成し、この共有鍵K_n、ＩＤｃ、ＩＤｓおよびＳＩＧＮＡＬ’$_{n,1}$を引数とするハッシュ関数ｈの関数値と、ＤＨ公開値g^{yn}と、ＳＩＧＮＡＬ’$_{n,1}$とをクライアント２０に対して送信する処理を実行する（ステップＰ３）。一方、受信データと記憶データとが一致せず、クライアント２０が正当でないと判定される場合には、クライアント２０からのアクセスを拒否して、当該認証処理を終了する。

【００９９】

次いで、ステップＰ４では、クライアント２０が、ＳＩＧＮＡＬ’$_{n,1}$を演算により求め、この演算結果と、サーバ１０から受信したＳＩＧＮＡＬ’$_{n,1}$との照合により、サーバ１０を識別し、識別できない場合には、通信を拒否する。識別できる場合には、サーバ１０から受信したＤＨ公開値g^{yn}と当該クライアント２０に予め記憶されたＤＨ公開値g^{xn}とからＤＨ共通鍵g^{xnyn}を共有鍵K_nとして生成するとともに、この共有鍵K_nを用いてハッシュ関数ｈの関数値を演算により求め、この演算結果と、サーバ１０から受信したハッシュ関数ｈの関数値との照合により、サーバ１０の正当性を判定する処理を実行する。

【０１００】

上記判定の結果、受信データと演算結果とが一致して、サーバ１０が正当であると判定される場合には、当該認証処理を終了して、次のデータ伝送処理に移行する。一方、受信データと演算結果とが一致せず、サーバ１０が正当でないと判定される場合には、サーバ１０へのアクセスを中止して、当該認証処理を終了する。

【０１０１】

なお、クライアント２０が共有鍵K_iを共有したことをサーバ１０側で確認する必要がある場合には、このステップＰ４でクライアント２０がサーバ１０の正当性を判定した後に、共有鍵K_n、ＩＤｃ、ＩＤｓを引数とするハッシュ関数ｈ

の関数値をサーバ１０に対して送信するようにすればよい。

【０１０２】

以上のように、この第２の実施形態によれば、前回のセッションで生成された共有鍵K_{i-1}（可変共有鍵）と今回のセッションにおける通信順序ｊとを引数とするハッシュ関数の関数値を求め、この関数値を当該セッションのｊ番目の通信にのみ有効なワンタイムＩＤ（ＳＩＧＮＡＬ）として用いるようにしたので、例えば、ｎ番目のセッションで生成した共有鍵K_nが第三者に漏れたとしても、セッション毎に共有鍵K_nが変化することとなるので、漏れた共有鍵K_nを用いて生成されたワンタイムＩＤ（$SIGNAL_{n+1,j}$、$SIGNAL'_{n+1,j}$）以外のワンタイムＩＤを予測できなくなる。すなわち、盗聴が困難で安全性に優れたワンタイムＩＤを生成することが可能になり、ワンタイムＩＤの将来にわたる安全性（ＰＦＳ）を実現することが可能になる。

【０１０３】

また、上記ワンタイムＩＤ（ＳＩＧＮＡＬ）を用いて、クライアント・サーバ間における認証を行うようにしたので、前述した第１の実施形態と同様、大量の計算要求・応答要求などによる計算量やメモリへのＤｏＳ攻撃を防止することができ、オープンなネットワーク環境下においても、ＩＤ情報の保護を図り、通信の安全性を向上させることができる。

【０１０４】

なお、ＤｏＳ攻撃を防止する手法の一つとして、クッキー（乱数）を用いた手法が一般に知られている。この方法によれば、ＩＰアドレスとクッキー生成者しか知らない秘密を組み合わせことにより、同一ＩＰアドレスからのＤｏＳ攻撃を防ぐことができる。これに対して、本実施形態のＳＩＧＮＡＬの場合には、ＤＨ共通鍵を知らない限り、次回有効となるＳＩＧＮＡＬを予測することができない。よって、毎回の通信にＳＩＧＮＡＬを利用することにより、クッキーと同様の効果が得られる。さらに、クッキーの場合はセッション中にＩＰアドレスが変わることを許さないが、ＳＩＧＮＡＬは変わっても良い。また、クッキーを用いた場合ＩＰアドレスを偽造したＤｏＳ攻撃を防ぐことができないが、ワンタイムＩＤではＩＰアドレスが関係ないためこのような攻撃も防ぐことができる。

【０１０５】

また、本実施形態において、例えば、クライアント２０がプロトコルの最初のメッセージを送り（ステップＰ１）、サーバ１０がそれに対応してＤＨ鍵交換の計算を行い（ステップＰ２）、２番目のメッセージを送った（ステップＰ３）場合を考える。もし、サーバ１０のメッセージが途中で消失、もしくは攻撃者に横取りされ、クライアント２０が受け取ることができなかった場合、クライアント２０はもう一度最初のメッセージを送信する必要がある。このとき、サーバ１０は正しいクライアント２０が通信を送りなおしてきたのか、攻撃者が最初のメッセージを読み取りリプレイ攻撃を行っているのか、判断することができない。そこで、クライアント２０はもう一度最初のメッセージを送りなおす場合、最初のチャレンジの際に送ったメッセージと同一の内容のものを送ることにし、サーバ１０も以前返信したメッセージのコピーをそのまま送ることにする。これにより、無駄なＤＨ鍵交換の計算を避けることができ、リプレイ攻撃によるＤｏＳ攻撃を防ぐことができる。

【０１０６】

なお、この実施形態では、前回のセッションで生成された共有鍵（ＤＨ共通鍵）K_{i-1} と今回のセッションにおける通信順序ｊとを引数とするハッシュ関数の関数値を求め、この関数値を当該セッションのｊ番目の通信にのみ有効なワンタイムＩＤ（ＳＩＧＮＡＬ）として生成するようにしたが、例えば、次のようにＳＩＧＮＡＬを生成することも可能である。

【０１０７】

【数４】

$$SS_j = h1(K_{i-1})$$

$$SIGNAL_{i,j} = hash(SS_i, j)$$

$$SIGNAL'_{i,j} = hash'(SS_i, j)$$

【０１０８】

上記ＳＩＧＮＡＬの定義式において、SS_i は（ｉ－１）番目のセッションで共有したＤＨ共通鍵K_{i-1} を引数とするハッシュ関数の関数値である。

また、この場合には、ｉ番目のセッションで用いられる認証用鍵をAK_i、暗

号化鍵をSK_iとして、これら鍵を、例えば、AK_i＝ｈ２（K_{i-1}）、SK_i＝ｈ３（K_{i-1}）という式から求めるようにしてもよい。なお、ｈ１、ｈ２、ｈ３は、衝突のない一方向性ハッシュ関数である。

【０１０９】

このようにSS_iから認証用鍵および暗号化鍵を生成する場合には、前述したステップＰ１において、クライアント２０が、ＩＤｃ、ＩＤｓ、ＤＨ公開値g^{xn}および$SIGNAL_{n,1}$を暗号化してサーバ１０に対して送信する際に、認証用鍵AK_nを用いるようにする。また、ステップＰ３において、サーバ１０がクライアント２０に対して送信するハッシュ関数ｈに、暗号化鍵SK_n、ＩＤｃ、ＩＤｓおよび$SIGNAL'_{n,1}$を引数とするハッシュ関数を用いるようにする。

【０１１０】

そうすることで、攻撃者が、仮に、SS_i、AK_i、SK_iの何れか１つの値を知ることができたとしても、その他の値を計算することはできない。よって、攻撃者がｉ番目のセッションにおいて正規ユーザに成りすまし、鍵交換を行うためには、AK_i、ＳＩＧＮＡＬ、正規ユーザのＩＤ情報（ＩＤｓ、ＩＤｃ）が必要となり、暗号通信するためには、SK_i ＳＩＧＮＡＬ、正規ユーザのＩＤ情報、通信回数の情報が必要となる。

また、ｎ番目のセッションにおけるクライアント２０のＤＨ公開値g^{xn}は、認証鍵AK_i（ｈ２（K_{i-1}））を用いて暗号化される。よって、AK_iを知らない攻撃者はg^{xn}を知ることができない。そのため、本方式で生成・共有されるDiffie-Hellman共通鍵は計算量的、かつ情報論的に安全である。

【０１１１】

[第３の実施形態]

前述した第１および第２の実施形態では、認証と同時にDiffie-Hellman鍵交換を行うようにしたが、この第３の実施形態では、Diffie-Hellman鍵交換を省略するようにしている。この第３の実施形態特有の部分以外は、第１の実施形態におけると同様である。この第３の実施形態において、第１の実施形態と同一部分には同一符号を付し、その説明を省略する。

【０１１２】

図６は本発明に係る認証方法の第３の実施形態を説明する図である。この第３の実施形態では、先ず、クライアント２０が、乱数Ｒｃ（第一の乱数）を生成するとともに、サーバ１０との間で予め共有化された共有鍵Ｋ１（第一の共有鍵）および乱数Ｒ０（初期乱数）を引数とする疑似乱数関数ｐｒｆ（Ｋ１、Ｒ０）の関数値を$ＳＩＧＮＡＬ_{c1}$（第一のワンタイムＩＤ）として求め（ステップＳ１１）、この$ＳＩＧＮＡＬ_{c1}$と、共有鍵Ｋ１で暗号化した乱数Ｒｃとをサーバ１０に対して送信する処理を実行する（ステップＳ１２）。

【０１１３】

次いで、サーバ１０が、乱数Ｒｓ（第二の乱数）を生成するとともに、共有鍵Ｋ１で復号した乱数Ｒｃと共有鍵Ｋ１とを引数とする疑似乱数関数ｐｒｆ（Ｋ１、Ｒｃ）の関数値を$ＳＩＧＮＡＬ_{s1}$（第二のワンタイムＩＤ）として求め（ステップＳ１３）、この$ＳＩＧＮＡＬ_{s1}$と、共有鍵Ｋ１で暗号化した乱数Ｒｓと、乱数Ｒ０＋Ｒｃ（乱数Ｒ０、Ｒｃを引数とする所定の演算結果、例えば、両者の排他的論理和など）とをクライアント２０に対して送信する処理を実行する（ステップＳ１４）。

【０１１４】

次いで、クライアント２０が、乱数Ｒｃと共有鍵Ｋ１に基づいて$ＳＩＧＮＡＬ_{s1}$を演算により求め、この演算結果とサーバ１０から受信した$ＳＩＧＮＡＬ_{s1}$との比較により、サーバ１０を識別するとともに、乱数Ｒ０＋Ｒｃの受信データと演算結果との比較により、サーバ１０の正当性を判定する処理を実行する（ステップＳ１５）。

【０１１５】

上記判定の結果、各々の受信データと演算結果とが一致して、サーバ１０が正当であると判定される場合には、クライアント２０が、乱数Ｒｃおよび乱数Ｒｓに基づいて共有鍵Ｋ２（第二の共有鍵）を生成するとともに、この共有鍵Ｋ２、乱数Ｒｓおよび乱数Ｒｃを引数とする疑似乱数関数ｐｒｆ（Ｋ２、Ｒｓ、Ｒｃ）の関数値を$ＳＩＧＮＡＬ_{c2}$（第三のワンタイムＩＤ）として求め、この$ＳＩＧＮＡＬ_{c2}$と、乱数Ｒｃ＋Ｒｓ（乱数Ｒｃ、Ｒｓを引数とする所定の演算結果）とをサーバ１０に対して送信する処理を実行する（ステップＳ１６）。一方、受信デ

ータと演算結果とが一致せず、サーバ１０が正当でないと判定される場合には、サーバ１０へのアクセスを中止して、当該認証処理を終了する。

【０１１６】

サーバ１０は、クライアント２０から$SIGNAL_{c2}$を受信すると、乱数Ｒｃおよび乱数Ｒｓに基づいて共有鍵Ｋ２を生成するとともに、共有鍵Ｋ２、乱数Ｒｓおよび乱数Ｒｃに基づいて$SIGNAL_{c2}$を演算により求め、この演算結果とクライアント２０から受信した$SIGNAL_{c2}$との比較により、クライアント２０を識別するとともに、乱数Ｒｃ＋Ｒｓの受信データと演算結果との比較により、クライアント２０の正当性を判定する処理を実行する（ステップＳ１７）。

【０１１７】

上記判定の結果、各々の受信データと演算結果とが一致して、クライアント２０が正当であると判定される場合には、当該認証処理を終了して、次のデータ伝送処理に移行する。一方、受信データと演算結果とが一致せず、クライアント２０が正当でないと判定される場合には、クライアント２０からのアクセスを拒否して、当該認証処理を終了する。

【０１１８】

以上のように、この第３の実施形態によれば、相互認証の過程で生成された乱数と、相互認証の過程で変化する共有鍵Ｋとを引数とする疑似乱数関数ｐｒｆの関数値をワンタイムＩＤとして用いるようにしたので、前述した第１の実施形態と同様、ワンタイムＩＤの安全性を高めることができ、迅速かつ安全な相互認証を実現することができる。

【０１１９】

［第４の実施形態］

前述した第３の実施形態では、ワンタイムＩＤ（ＳＩＧＮＡＬ）の生成に用いる共有鍵を相互認証の過程で変化させるようにしたが、この第４の実施形態では、上記共有鍵を固定するようにしている。

【０１２０】

すなわち、この第４の実施形態では、図７に示すように、先ず、クライアント２０が、乱数Ｒｃ（第一の乱数）を生成するとともに、サーバ１０との間で予め

共有化された共有鍵Ｋおよび乱数Ｒ０（初期乱数）を引数とする疑似乱数関数ｐｒｆ（Ｋ、Ｒ０）の関数値をＳＩＧＮＡＬ$_{c1}$（第一のワンタイムＩＤ）として求め（ステップＳ２１）、このＳＩＧＮＡＬ$_{c1}$と、共有鍵Ｋで暗号化した乱数Ｒｃとをサーバ１０に対して送信する処理を実行する（ステップＳ２２）。

【０１２１】

次いで、サーバ１０が、乱数Ｒｓ（第二の乱数）を生成するとともに、共有鍵Ｋで復号した乱数Ｒｃと共有鍵Ｋとを引数とする疑似乱数関数ｐｒｆ（Ｋ、Ｒｃ）の関数値をＳＩＧＮＡＬ$_{s1}$（第二のワンタイムＩＤ）として求め（ステップＳ２３）、このＳＩＧＮＡＬ$_{s1}$と、共有鍵Ｋで暗号化した乱数Ｒｓと、乱数Ｒ０＋Ｒｃ（乱数Ｒ０、Ｒｃを引数とする所定の演算結果）とをクライアント２０に対して送信する処理を実行する（ステップＳ２４）。

【０１２２】

次いで、クライアント２０が、乱数Ｒｃおよび共有鍵Ｋに基づいてＳＩＧＮＡＬ$_{s1}$を演算により求め、この演算結果とサーバ１０から受信したＳＩＧＮＡＬ$_{s1}$との比較により、サーバ１０を識別するとともに、乱数Ｒ０＋Ｒｃの受信データと演算結果との比較により、サーバ１０の正当性を判定する処理を実行する（ステップＳ２５）。

【０１２３】

上記判定の結果、各々の受信データと演算結果とが一致して、サーバ１０が正当であると判定される場合には、クライアント２０が、乱数Ｒｃ、乱数Ｒｓおよび共有鍵Ｋを引数とする疑似乱数関数ｐｒｆ（Ｋ、Ｒｓ、Ｒｃ）の関数値をＳＩＧＮＡＬ$_{c2}$（第三のワンタイムＩＤ）として求め、このＳＩＧＮＡＬ$_{c2}$と、乱数Ｒｃ＋Ｒｓ（乱数Ｒｃ、Ｒｓを引数とする所定の演算結果）とをサーバ１０に対して送信する処理を実行する（ステップＳ２６）。一方、受信データと演算結果とが一致せず、サーバ１０が正当でないと判定される場合には、サーバ１０へのアクセスを中止して、当該認証処理を終了する。

【０１２４】

サーバ１０は、クライアント２０からＳＩＧＮＡＬ$_{c2}$を受信すると、乱数Ｒｃ、乱数Ｒｓおよび共有鍵Ｋに基づいてＳＩＧＮＡＬ$_{c2}$を演算により求め、この演

算結果とクライアント２０から受信したＳＩＧＮＡＬ$_{c2}$との比較により、クライアント２０を識別するとともに、乱数Ｒｃ＋Ｒｓの受信データと演算結果との比較により、クライアント２０の正当性を判定する処理を実行する（ステップＳ２７）。

【０１２５】

上記判定の結果、各々の受信データと演算結果とが一致して、クライアント２０が正当であると判定される場合には、当該認証処理を終了して、次のデータ伝送処理に移行する。一方、受信データと演算結果とが一致せず、クライアント２０が正当でないと判定される場合には、クライアント２０からのアクセスを拒否して、当該認証処理を終了する。

【０１２６】

以上のように、この第４の実施形態によれば、相互認証の過程で生成された乱数と共有鍵Ｋとを引数とする疑似乱数関数ｐｒｆの関数値をワンタイムＩＤとして用いるようにしたため、例えば、共有鍵Ｋが第三者に漏れたとしても、乱数によって疑似乱数関数ｐｒｆの関数値が相互認証の過程で順次変化することとなるので、相互認証の過程で生成される乱数がわからない限り、ワンタイムＩＤを予測することができない。したがって、前述した第１～第３の実施形態と同様、ワンタイムＩＤの安全性を高めることができ、迅速かつ安全な相互認証を実現することができる。

【０１２７】

［第５の実施形態］

図８は本発明に係る認証方法の第５の実施形態を説明する図である。この第５の実施形態では、先ず、クライアント２０が、乱数Ｒ$_{ci}$（第一の乱数）を生成するとともに、サーバ１０との間で予め共有化された共有鍵Ｋ$_{i}$、乱数Ｒ$_{ci-1}$（第一の記憶乱数）および乱数Ｒ$_{si-1}$（第二の記憶乱数）を引数とする疑似乱数関数ｐｒｆ（Ｋ$_{i}$、Ｒ$_{ci-1}$、Ｒ$_{si-1}$）の関数値をＳＩＧＮＡＬ$_{ci}$（第一のワンタイムＩＤ）として求める処理を実行する（ステップＳ３１）。

【０１２８】

なお、Ｒ$_{ci}$はｉ番目のセッションでクライアント２０により生成された乱数、

R_{si}はｉ番目のセッションでサーバ１０により生成された乱数、K_i はｉ番目のセッションで使用する可変共有鍵をそれぞれ示している。また、前回（ｉ－１番目）のセッションで生成された乱数R_{ci-1}、R_{si-1}は、サーバ１０とクライアント２０の各記憶装置１３、２３の記憶領域に格納されており、これら乱数R_{ci-1}、R_{si-1}に基づいて、共有鍵K_i が生成されるようになっている。

【０１２９】

そして、クライアント２０は、ＳＩＧＮＡＬ$_{ci}$を生成した後、ＩＤｃ（クライアントＩＤ）、ＩＤｓ（サーバＩＤ）および乱数R_{ci}を共有鍵K_i で暗号化した暗号化データE_{Ki}（ＩＤｃ、ＩＤｓ、R_{ci}）と、ＳＩＧＮＡＬ$_{ci}$とをサーバ１０に対して送信する処理を実行する（ステップＳ３２）。

【０１３０】

サーバ１０は、クライアント２０からＳＩＧＮＡＬ$_{ci}$を受信すると、共有鍵K_i、乱数R_{ci-1}および乱数R_{si-1}に基づいてＳＩＧＮＡＬ$_{ci}$を演算により求め、この演算結果とクライアント２０から受信したＳＩＧＮＡＬ$_{ci}$との比較により、クライアント２０を識別し、識別できない場合には、通信を拒否する。識別できる場合には、共有鍵K_i を用いて暗号化データE_{Ki}（ＩＤｃ、ＩＤｓ、R_{ci}）を復号し、この復号したデータに含まれるＩＤｃおよびＩＤｓに基づいて、クライアント２０の正当性を判定する処理を実行する。

【０１３１】

上記判定の結果、受信データと、サーバ１０に予め格納された記憶データとが一致して、クライアント２０が正当であると判定される場合には、乱数R_{si}（第二の乱数）を生成するとともに、乱数R_{ci}、乱数R_{si-1}および共有鍵K_i を引数とする疑似乱数関数ｐｒｆ（K_i 、R_{ci}、R_{si-1}）の関数値をＳＩＧＮＡＬ$_{si}$（第二のワンタイムＩＤ）として求める。そして、乱数R_{ci-1}、R_{si-1}を格納していた記憶領域に、乱数R_{ci}、R_{si}をそれぞれ格納するとともに、これら乱数R_{ci}、R_{si}に基づき共有鍵K_{i+1} を生成して記憶する処理を実行する（ステップＳ３３）。

【０１３２】

次いで、サーバ１０は、ＩＤｃ、ＩＤｓおよび乱数R_{si}を共有鍵K_i で暗号化

した暗号化データE_{Ki}（ＩＤｓ、ＩＤｃ、R_{si}）と、$SIGNAL_{si}$とをクライアント２０に対して送信する処理を実行する（ステップＳ３４）。

一方、受信データと記憶データとが一致せず、クライアント２０が正当でないと判定される場合には、クライアント２０からのアクセスを拒否して、当該認証処理を終了する。

【０１３３】

クライアント２０は、サーバ１０から$SIGNAL_{si}$を受信すると、共有鍵K_i、乱数R_{ci}および乱数R_{si-1}に基づいて$SIGNAL_{si}$を演算により求め、この演算結果とクライアント２０から受信した$SIGNAL_{si}$との比較により、サーバ１０を識別し、識別できない場合には、通信を拒否する。一方、識別できる場合には、共有鍵K_iを用いて暗号化データE_{Ki}（ＩＤｓ、ＩＤｃ、R_{si}）を復号し、この復号したデータに含まれるＩＤｃおよびＩＤｓに基づいて、サーバ１０の正当性を判定する処理を実行する。サーバ１０を識別できる場合、通信相手を特定できるだけでなく、サーバ１０が乱数R_{ci}を受け取ったことも確認することができる。

【０１３４】

そして、上記判定の結果、受信データと、クライアント２０に予め格納された記憶データとが一致して、サーバ１０が正当であると判定される場合には、乱数R_{ci-1}、R_{si-1}を格納していた記憶領域に、乱数R_{ci}、R_{si}をそれぞれ格納して、これら乱数R_{ci}、R_{si}に基づき共有鍵K_{i+1}を生成・記憶した後（ステップＳ３５）、当該認証処理を終了して、次のデータ伝送処理に移行する。一方、受信データと記憶データとが一致せず、サーバ１０が正当でないと判定される場合には、サーバ１０からのアクセスを拒否して、当該認証処理を終了する。

【０１３５】

以上のように、この第５の実施形態によれば、前述した第３の実施形態と同様の作用・効果が得られるのに加えて、ＩＤｃ、ＩＤｓおよび乱数R_{si}を共有鍵K_iで暗号化した暗号化データE_{Ki}（ＩＤｓ、ＩＤｃ、R_{si}）を通信相手に送信するようにしたことにより、例えば、攻撃者によって暗号化データが書き換えられた場合においても、暗号化データに含まれるＩＤ情報（ＩＤｓ、ＩＤｃ）が正し

く復号されないために、このデータを受け取ったサーバ１０またはクライアント２０は、送られてきた暗号化データが誤ったものであることを容易に検出でき、乱数を受け取らずに廃棄することが可能となる。また、$SIGNAL_{ci}$の値が他の複数のクライアントと重複した場合においても、暗号化データに含まれるＩＤ情報（ＩＤｓ、ＩＤｃ）を参照することによって、通信相手を容易に特定することができる。

【０１３６】

さらに、この第５の実施形態によれば、通信相手がサーバ・クライアントのＩＤ情報（ＩＤｓ、ＩＤｃ）を正しく暗号化できているか否かを確認することによって、通信相手の正当性を判定するようにしたので、前述した第３の実施形態では３回必要であった通信回数を２回に低減することが可能となり、より効率的な認証が可能となる。

【０１３７】

［第６の実施形態］

図９は本発明に係る認証方法の第６の実施形態を説明する図である。この第６の実施形態では、先ず、クライアント２０が、乱数R_{ci}（第一の乱数）を生成するとともに、サーバ１０との間で予め共有化された固定共有鍵Ｋ、乱数R_{ci-1}（第一の記憶乱数）および乱数R_{si-1}（第二の記憶乱数）を引数とする疑似乱数関数ｐｒｆ（Ｋ、R_{ci-1}、R_{si-1}）の関数値を$SIGNAL_{ci}$（第一のワンタイムＩＤ）として求める処理を実行する（ステップＳ４１）。

【０１３８】

なお、R_{ci}はｉ番目のセッションでクライアント２０により生成された乱数、R_{si}はｉ番目のセッションでサーバ１０により生成された乱数をそれぞれ示している。また、前回（ｉ－１番目）のセッションで生成された乱数R_{ci-1}、R_{si-1}は、サーバ１０とクライアント２０の各記憶装置１３、２３の記憶領域に格納されている。

【０１３９】

そして、クライアント２０は、$SIGNAL_{ci}$を生成した後、ＩＤｃ（クライアントＩＤ）、ＩＤｓ（サーバＩＤ）および乱数R_{ci}を共有鍵Ｋで暗号化した暗

号化データE_K（ＩＤｃ、ＩＤｓ、R_{ci}）と、$SIGNAL_{ci}$とをサーバ１０に対して送信する処理を実行する（ステップＳ４２）。

【０１４０】

サーバ１０は、クライアント２０から$SIGNAL_{ci}$を受信すると、共有鍵Ｋ、乱数R_{ci-1}および乱数R_{si-1}に基づいて$SIGNAL_{ci}$を演算により求め、この演算結果とクライアント２０から受信した$SIGNAL_{ci}$との比較により、クライアント２０を識別し、識別できない場合には、通信を拒否する。識別できる場合には、共有鍵Ｋを用いて暗号化データE_K（ＩＤｃ、ＩＤｓ、R_{ci}）を復号し、この復号したデータに含まれるＩＤｃおよびＩＤｓに基づいて、クライアント２０の正当性を判定する処理を実行する。

【０１４１】

上記判定の結果、受信データと、サーバ１０に予め格納された記憶データとが一致して、クライアント２０が正当であると判定される場合には、乱数R_{si}（第二の乱数）を生成するとともに、乱数R_{ci}、乱数R_{si-1}および共有鍵Ｋを引数とする疑似乱数関数ｐｒｆ（Ｋ、R_{ci}、R_{si-1}）の関数値を$SIGNAL_{si}$（第二のワンタイムＩＤ）として求める。そして、乱数R_{ci-1}、R_{si-1}を格納していた記憶領域に、乱数R_{ci}、R_{si}をそれぞれ格納する処理を実行する（ステップＳ４３）。

【０１４２】

次いで、サーバ１０は、ＩＤｃ、ＩＤｓおよび乱数R_{si}を共有鍵Ｋで暗号化した暗号化データE_K（ＩＤｓ、ＩＤｃ、R_{si}）と、$SIGNAL_{si}$とをクライアント２０に対して送信する処理を実行する（ステップＳ４４）。

一方、受信データと記憶データとが一致せず、クライアント２０が正当でないと判定される場合には、クライアント２０からのアクセスを拒否して、当該認証処理を終了する。

【０１４３】

クライアント２０は、サーバ１０から$SIGNAL_{si}$を受信すると、共有鍵Ｋ、乱数R_{ci}および乱数R_{si-1}に基づいて$SIGNAL_{si}$を演算により求め、この演算結果とクライアント２０から受信した$SIGNAL_{si}$との比較により、サー

バ１０を識別し、識別できない場合には、通信を拒否する。一方、識別できる場合には、共有鍵Ｋを用いて暗号化データE_K（ＩＤｓ、ＩＤｃ、R_{si}）を復号し、この復号したデータに含まれるＩＤｃおよびＩＤｓに基づいて、サーバ１０の正当性を判定する処理を実行する。サーバ１０を識別できる場合、通信相手を特定できるだけでなく、サーバ１０が乱数R_{ci}を受け取ったことも確認することができる。

【０１４４】

そして、上記判定の結果、受信データと、クライアント２０に予め格納された記憶データとが一致して、サーバ１０が正当であると判定される場合には、乱数R_{ci-1}、R_{si-1}を格納していた記憶領域に、乱数R_{ci}、R_{si}をそれぞれ格納して、これら乱数R_{ci}、R_{si}に基づき共有鍵Ｋを生成・記憶した後（ステップＳ４５）、当該認証処理を終了して、次のデータ伝送処理に移行する。一方、受信データと記憶データとが一致せず、サーバ１０が正当でないと判定される場合には、サーバ１０からのアクセスを拒否して、当該認証処理を終了する。

【０１４５】

以上のように、この第６の実施形態によれば、前述した第４の実施形態と同様の作用・効果が得られるのに加えて、例えば、攻撃者によって暗号化データが書き換えられた場合においても、データを受け取ったサーバ１０またはクライアント２０は、送られてきた暗号化データが誤ったものであることを容易に検出でき、乱数を受け取らずに廃棄することが可能となる。また、$SIGNAL_{ci}$の値が他の複数のクライアントと重複した場合においても、暗号化データに含まれるＩＤ情報（ＩＤｓ、ＩＤｃ）を参照することによって、通信相手を容易に特定することができる。さらに、この第６の実施形態によれば、前述した第４の実施形態では３回必要であった通信回数を２回に低減することが可能となり、より効率的な認証が可能となる。

【０１４６】

[第７の実施形態]

この第７の実施形態では、ワンタイムＩＤを用いたリプレイ攻撃の防止方法について説明する。リプレイ攻撃とは、過去に正式な通信者が送信したときに有効

であった通信情報を攻撃者（第三者）が盗聴し、再利用する攻撃のことである。

【０１４７】

先ず、ＯＳＰＡ（Optimal Strong Password Authentication）と呼ばれるパスワードを利用した認証方式（Chun-Li LIN, Hung-Min SUN, Tzonelih HWANG, Attacks and Solutions on Strong- Password Authentication, IEICE TRANS. COMMUN., VOL.E84-B, NO.9, September 2001.）について、図１０に基づいて説明する。

当該認証に先立って、クライアント２０には、ハッシュ関数ｈおよびパスワードＰが予め記憶保持されており、サーバ１０には、ハッシュ関数ｈ、セッション回数ｎ、ＩＤｃ（クライアントＩＤ）および検証用情報h^2（Ｐ＠ｎ）が予め記憶保持されている。検証用情報h^2（Ｐ＠ｎ）は、クライアント２０の正当性を検証するための情報であって、パスワードＰと通信回数ｎの排他的論理和を用いてハッシュ関数ｈにより生成された情報である。なお、h^2（Ｐ＠ｎ）は、ハッシュ関数ｈの計算を２回行うこと、つまりｈ（ｈ（Ｐ＠ｎ））を示しており、この数式中の＠は排他的論理和を示している。

【０１４８】

この認証方式では、先ず、クライアント２０がサーバ１０に対してＩＤｃを送信する（ステップＳ５１）。

サーバ１０は、クライアント２０からＩＤｃを受信すると、この受信したＩＤｃと、予め記憶しているＩＤｃとの比較により、クライアント２０を識別し、識別できない場合には、通信を拒否する。識別できる場合には、サーバ１０に対してセッション回数ｎを送信する（ステップＳ５２）。

【０１４９】

クライアント２０は、サーバ１０からセッション回数ｎを受信すると、この受信したセッション回数ｎ、予め記憶しているハッシュ関数ｈおよびパスワードＰを用いて、第１～第３の認証用情報Ｃ１、Ｃ２、Ｃ３を生成し（ステップＳ５３）、これらＣ１、Ｃ２、Ｃ３をサーバ１０に対して送信する（ステップＳ５４）。ここで、Ｃ１＝ｈ（Ｐ＠ｎ）＠h^2（Ｐ＠ｎ）、Ｃ２＝h^2（Ｐ＠（ｎ＋１））＠ｈ（Ｐ＠ｎ）、Ｃ３＝h^3（Ｐ＠（ｎ＋１））である。

【０１５０】

サーバ１０は、クライアント２０からＣ１、Ｃ２、Ｃ３を受信すると、先ず、受信したＣ１≠Ｃ２であることを確認する。これは、Ｃ１＝ｈ（Ｐ＠ｎ）＠h^2（Ｐ＠ｎ）、Ｃ２＝ｈ（Ｐ＠ｎ）＠h^2（Ｐ＠ｎ）、Ｃ３＝h^3（Ｐ＠ｎ）と計算して送った場合においても、サーバ１０がクライアント２０を認証してしまい、次の検証用情報として、h^2（Ｐ＠（ｎ＋１））ではなくh^2（Ｐ＠ｎ）を記憶してしまう不具合が発生する可能性があることから、このような不具合の発生を防ぐために行われるものである。

【０１５１】

次いで、サーバ１０は、Ｃ１、Ｃ２から、ｈ（Ｐ＠ｎ）、h^2（Ｐ＠（ｎ＋１））を演算により求める。すなわち、受信したＣ１と、予め記憶している検証用情報h^2（Ｐ＠ｎ）との排他的論理和を求めることでｈ（Ｐ＠ｎ）を導き出し、このｈ（Ｐ＠ｎ）と受信したＣ２との排他的論理和を求めることでh^2（Ｐ＠（ｎ＋１））を導き出す。

【０１５２】

次いで、予め記憶しているハッシュ関数ｈを用いて、求めたｈ（Ｐ＠ｎ）からｈ（ｈ（Ｐ＠ｎ））を計算し、このｈ（ｈ（Ｐ＠ｎ））が、予め記憶している検証用情報h^2（Ｐ＠ｎ）と一致するか否かを検証する。同時に、求めたh^2（Ｐ＠（ｎ＋１））から上記ハッシュ関数ｈを用いてｈ（h^2（Ｐ＠（ｎ＋１））を計算し、このｈ（h^2（Ｐ＠（ｎ＋１））が、受信したＣ３と一致するか否かを検証する（ステップＳ５５）。

【０１５３】

これら検証の結果、何れもが一致して、クライアント２０が正当であると判定される場合には、検証用情報をh^2（Ｐ＠ｎ）からh^2（Ｐ＠（ｎ＋１））に更新し、セッション回数をｎからｎ＋１に更新した後、クライアント２０からのアクセスを承諾して、当該認証処理を終了する。一方、上記検証の結果、少なくとも何れか一方が一致せず、クライアント２０が正当でないと判定される場合には、クライアント２０からのアクセスを拒否して、当該認証処理を終了する。

【０１５４】

上記認証方式によれば、盗聴者に対して安全な認証を行うことができるとともに、セッション毎に検証用情報をh^2（Ｐ＠ｎ）からh^2（Ｐ＠（ｎ＋１））へと更新することができるという利点がある。

しかしながら、上記認証方式にあっては、一度使用された認証情報Ｃ１、Ｃ２、Ｃ３をもう一度利用することによるリプレイ攻撃を防止することができないという問題点があった。

【０１５５】

そこで、本発明者等は、このような問題点を解決する認証方式として、次のような認証方式を開発した。

図１１は、本発明に係る認証方法の第７の実施形態を説明する図である。この図１１に示すように、クライアント２０に、ハッシュ関数ｈおよびパスワードＰが予め記憶保持され、サーバ１０に、ハッシュ関数ｈ、セッション回数ｎ、ＩＤｃおよび検証用情報h^2（Ｐ＠ｎ）が予め記憶保持されている場合には、先ず、クライアント２０がサーバ１０に対してＩＤｃを送信する（ステップＳ６１）。

【０１５６】

サーバ１０は、クライアント２０からＩＤｃを受信すると、この受信したＩＤｃと、予め記憶しているＩＤｃとの比較により、クライアント２０を識別し、識別できない場合には、通信を拒否する。識別できる場合には、サーバ１０に対してセッション回数ｎを送信する（ステップＳ６２）。

【０１５７】

クライアント２０は、サーバ１０からセッション回数ｎを受信すると、この受信したセッション回数ｎ、予め記憶しているハッシュ関数ｈおよびパスワードＰを用いて、第１～第３の認証用情報Ｃ１、Ｃ２、Ｃ３、$SIGNAL_n$を生成し（ステップＳ６３）、これらＣ１、Ｃ２、Ｃ３、$SIGNAL_n$をサーバ１０に対して送信する（ステップＳ６４）。ここで、Ｃ１＝ｈ（Ｐ＠ｎ）＠h^2（Ｐ＠ｎ）、Ｃ２＝h^2（Ｐ＠（ｎ＋１））＠ｈ（Ｐ＠ｎ）、Ｃ３＝h^3（Ｐ＠（ｎ＋１））、$SIGNAL_n$＝ｈ（h^2（Ｐ＠ｎ）、ｎ）である。すなわち、ｎ番目のセッションで使用するワンタイムＩＤである$SIGNAL_n$が、検証用情報ｈ

^{2}（Ｐ＠ｎ）およびセッション回数ｎを引数とするハッシュ関数ｈの関数値となっている。

【０１５８】

サーバ１０は、クライアント２０からＣ１、Ｃ２、Ｃ３、$ＳＩＧＮＡＬ_n$を受信すると、先ず、予め記憶している検証用情報h^2（Ｐ＠ｎ）とセッション回数ｎとに基づいて$ＳＩＧＮＡＬ_n$を演算により求め、この演算結果とクライアント２０から受信した$ＳＩＧＮＡＬ_n$との比較により、クライアント２０を識別し、識別できない場合には、通信を拒否する。識別できる場合には、受信したＣ１≠Ｃ２であることを確認した後、Ｃ１およびＣ２からｈ（Ｐ＠ｎ）、h^2（Ｐ＠（ｎ＋１））を演算により求める。

【０１５９】

次いで、サーバ１０は、予め記憶しているハッシュ関数ｈを用いて、求めたｈ（Ｐ＠ｎ）からｈ（ｈ（Ｐ＠ｎ））を計算し、このｈ（ｈ（Ｐ＠ｎ））が、予め記憶している検証用情報h^2（Ｐ＠ｎ）と一致するか否かを検証する。同時に、求めたh^2（Ｐ＠（ｎ＋１））から上記ハッシュ関数ｈを用いてｈ（h^2（Ｐ＠（ｎ＋１））を計算し、このｈ（h^2（Ｐ＠（ｎ＋１））が、受信したＣ３と一致するか否かを検証する（ステップＳ６５）。

【０１６０】

これら検証の結果、何れもが一致して、クライアント２０が正当であると判定される場合には、検証用情報をh^2（Ｐ＠ｎ）からh^2（Ｐ＠（ｎ＋１））に更新し、セッション回数をｎからｎ＋１に更新した後、クライアント２０からのアクセスを承諾して、当該認証処理を終了する。一方、上記検証の結果、少なくとも何れか一方が一致せず、クライアント２０が正当でないと判定される場合には、クライアント２０からのアクセスを拒否して、当該認証処理を終了する。

【０１６１】

上記認証方式によれば、検証用情報であるh^2（Ｐ＠ｎ）が攻撃者に知られる虞がないので、次のセッションのＳＩＧＮＡＬが攻撃者に予測されることはない。しかも、ＳＩＧＮＡＬは他のセッションで使用できないので、攻撃者によるリプレイ攻撃を効果的に防ぐことができる。

【０１６２】

なお、図１２に示すように、ハッシュ関数ｈおよびパスワードＰに加えて、予めセッション回数ｎもクライアント２０に記憶保持されている場合には、前述したステップＳ６１、Ｓ６２の処理を省略することが可能である。したがって、この場合には、ＩＤ情報（ＩＤｃ）の盗聴に対する保護を図りつつも、攻撃者によるリプレイ攻撃を効果的に防ぐことが可能である。

【０１６３】

なお、以上の各実施形態においては、複数の装置間の認証にワンタイムＩＤを用いるようにしたが、一装置内の複数のアプリケーション間の認証にワンタイムＩＤを用いることも可能である。また、以上の各実施形態においては、本発明に係る認証方法をクライアントサーバシステムに適用した場合について例示したが、これに限られるものではなく、例えば、Ｐ２Ｐ（Peer to Peer）システムに本発明に係る認証方法を適用することも可能である。

【０１６４】

また、本発明に係る認証方法をユーザによるアクセス毎に利用することも可能であり、その場合には、ユーザによるパスワードの入力を促して、パスワード、若しくはパスワードから生成した値（ワンタイムパスワードを含む。）をワンタイムＩＤとともに認証用のデータとして用いることが可能である。

【０１６５】

【発明の効果】

以上説明したように、請求項１～請求項６の何れかに記載の発明によれば、盗聴が困難で安全性に優れたワンタイムＩＤを生成することが可能となり、ワンタイムＩＤの将来にわたる安全性（ＰＦＳ）を実現することが可能となる。

【０１６６】

請求項７～請求項３５の何れかに記載の発明によれば、請求項１～請求項６の何れかに記載のワンタイムＩＤの生成方法により生成されたワンタイムＩＤを用いて、装置間（クライアント・サーバ間）の認証を行うようにしたので、第三者が送信者・受信者を特定できなくなる一方で、正当な送信者・受信者であればワンタイムＩＤを識別情報として把握できるようになる。

したがって、ＤｏＳ攻撃やなりすまし等に対する耐性を強化することができ、オープンなネットワーク環境下においても、ＩＤ情報の保護を図り、通信の安全性を向上させることができる。また、リモートアクセスが可能になり、利便性の向上を図ることもできる。

【０１６７】

請求項８に記載の発明によれば、従来の鍵交換・認証方式では３回必要であった通信回数を２回に低減することが可能になり、迅速かつ安全な認証および鍵交換を実現することが可能になる。

【図面の簡単な説明】

【図１】

本発明に係る認証システムの一実施形態を示す概略構成図である。

【図２】

図１のサーバの概略構成を示すブロック図である。

【図３】

図１のクライアントの概略構成を示すブロック図である。

【図４】

本発明に係る認証方法の第１の実施形態を説明する図である。

【図５】

本発明に係る認証方法の第２の実施形態を説明する図である。

【図６】

本発明に係る認証方法の第３の実施形態を説明する図である。

【図７】

本発明に係る認証方法の第４の実施形態を説明する図である。

【図８】

本発明に係る認証方法の第５の実施形態を説明する図である。

【図９】

本発明に係る認証方法の第６の実施形態を説明する図である。

【図１０】

ＯＳＰＡと呼ばれる従来の認証方法を説明する図である。

【図１１】

本発明に係る認証方法の第７の実施形態を説明する図である。

【図１２】

図１１の変形例を説明する図である。

【図１３】

Ｐ－ＳＩＧＭＡと呼ばれる従来の認証方法を説明する図である。

【符号の説明】

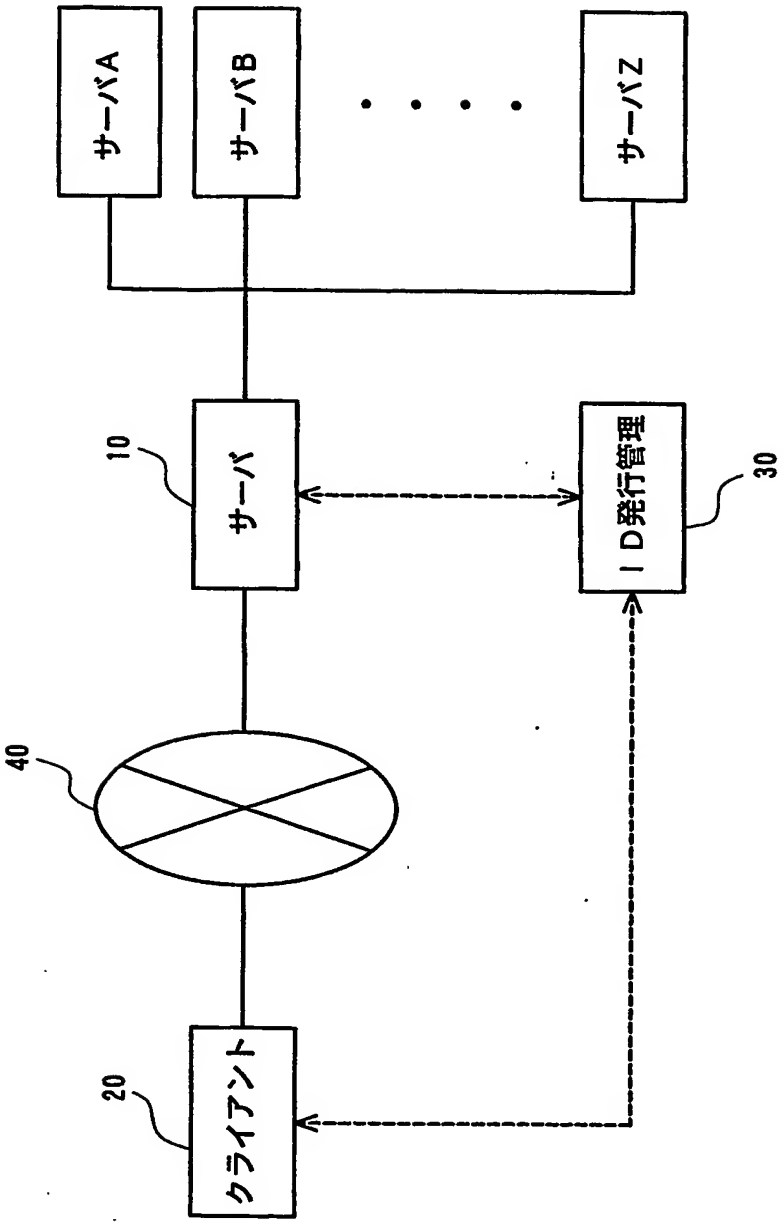
１０　サーバ（第二装置）

２０　クライアント（第一装置）

１１、２１　ＣＰＵ

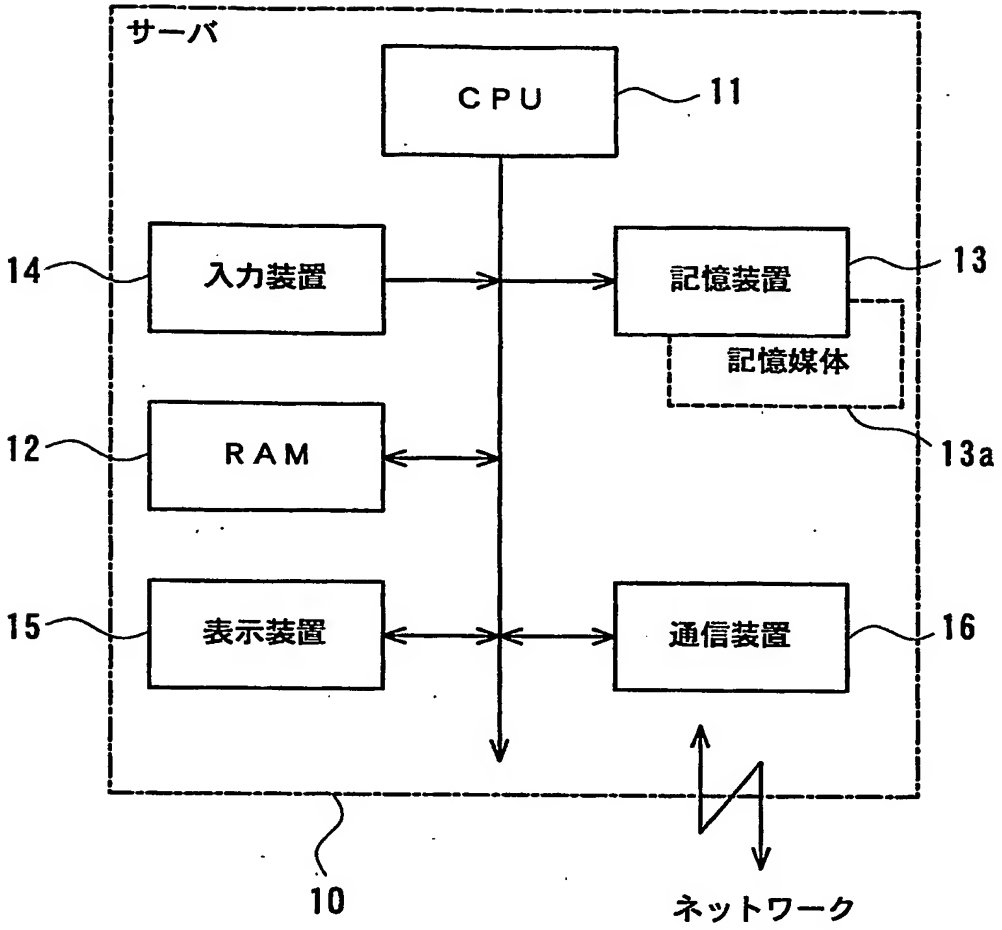
【書類名】　　図面

【図１】

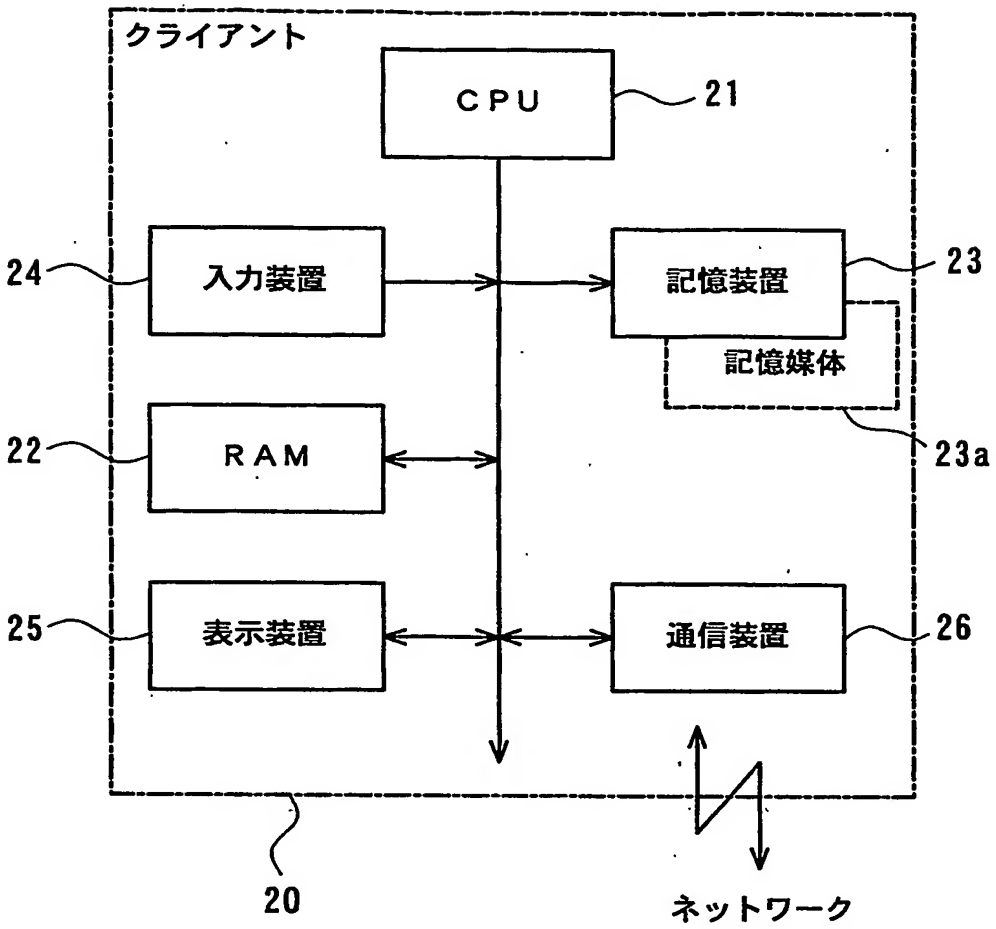


出証特２００３－３０４８４０７

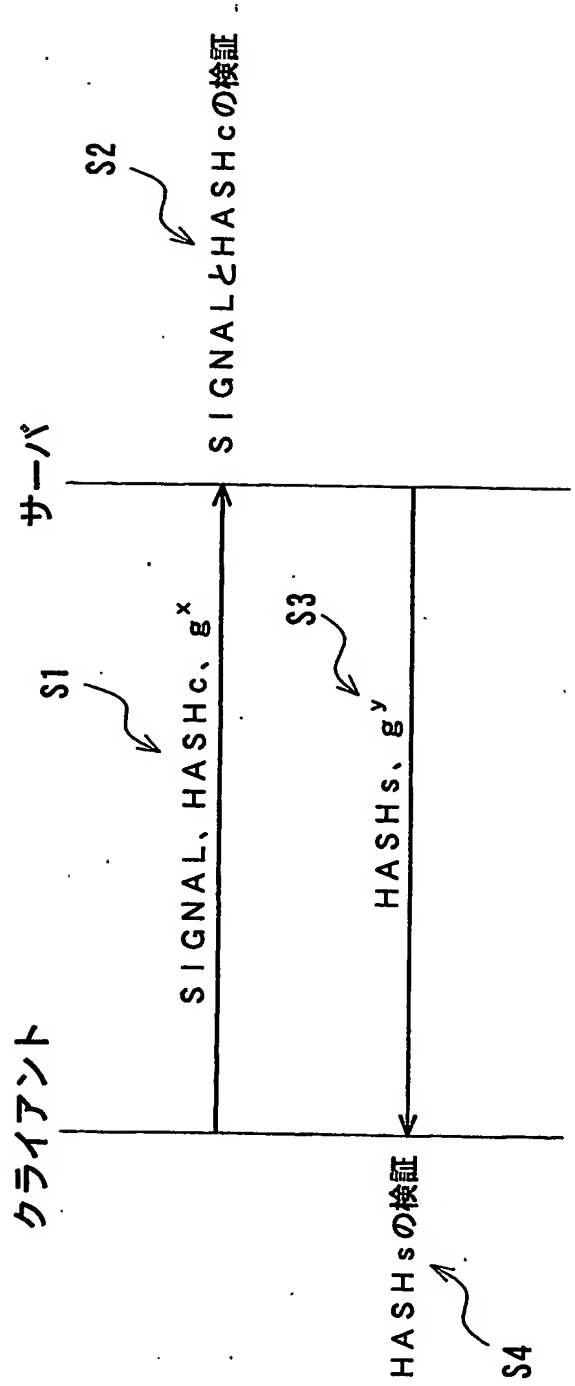
【図２】


サーバ
CPU
11
14
入力装置
記憶装置
13
記憶媒体
12
RAM
13a
15
表示装置
通信装置
16
10
ネットワーク

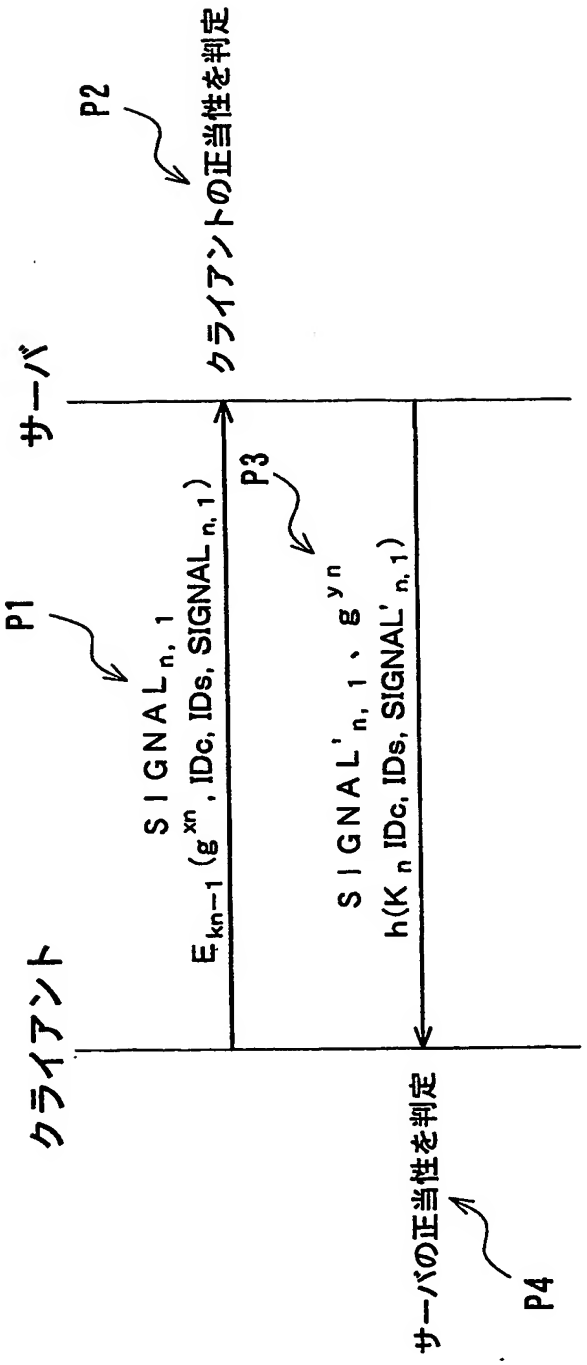
【図３】


クライアント
CPU
21
24
入力装置
記憶装置
23
記憶媒体
22
RAM
23a
25
表示装置
通信装置
26
20
ネットワーク

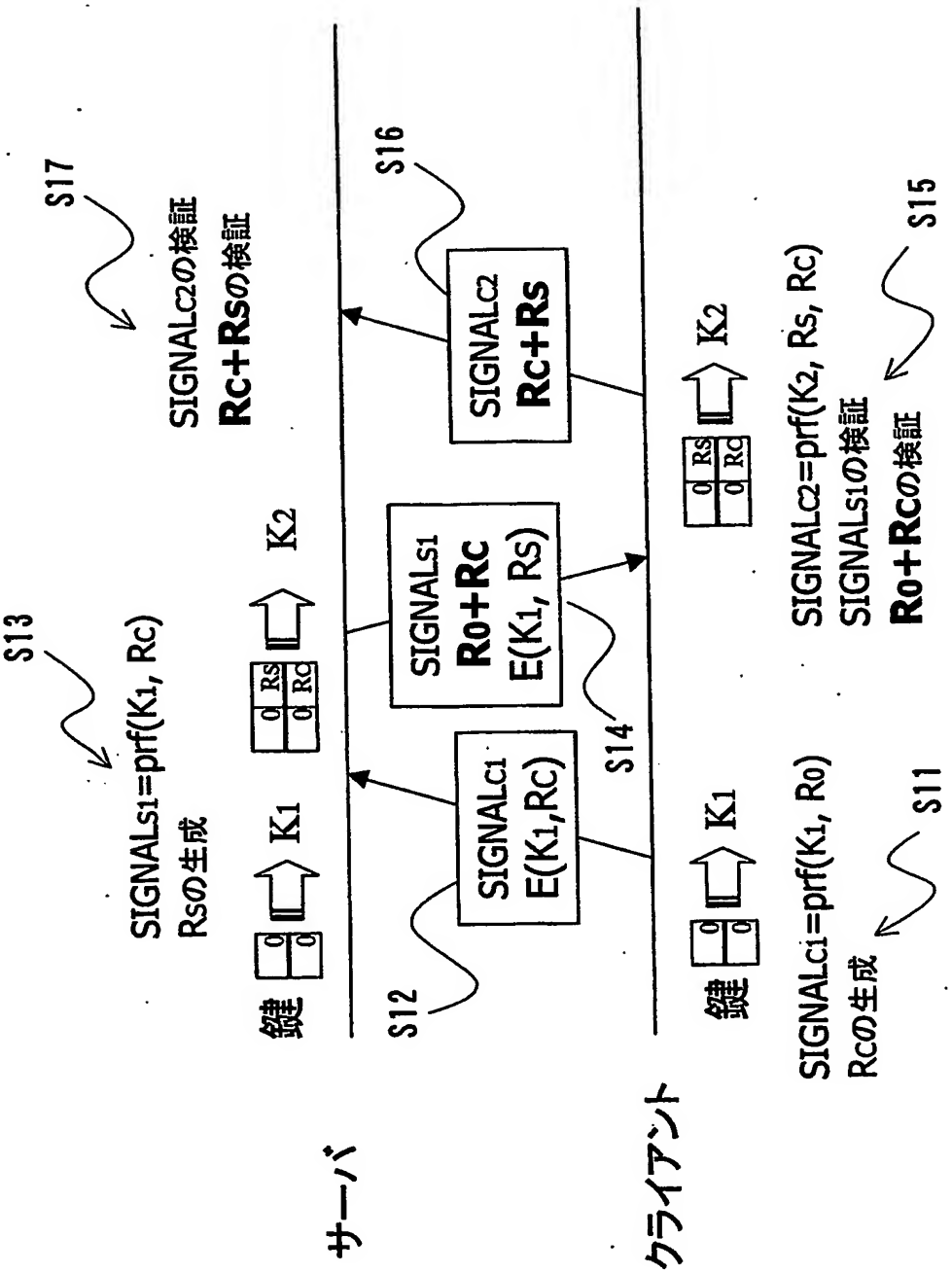
【図４】


サーバ
クライアント
S1
SIGNAL、HASHc、g^x
S2
SIGNALとHASHcの検証
S3
HASHs、g^y
HASHsの検証
S4

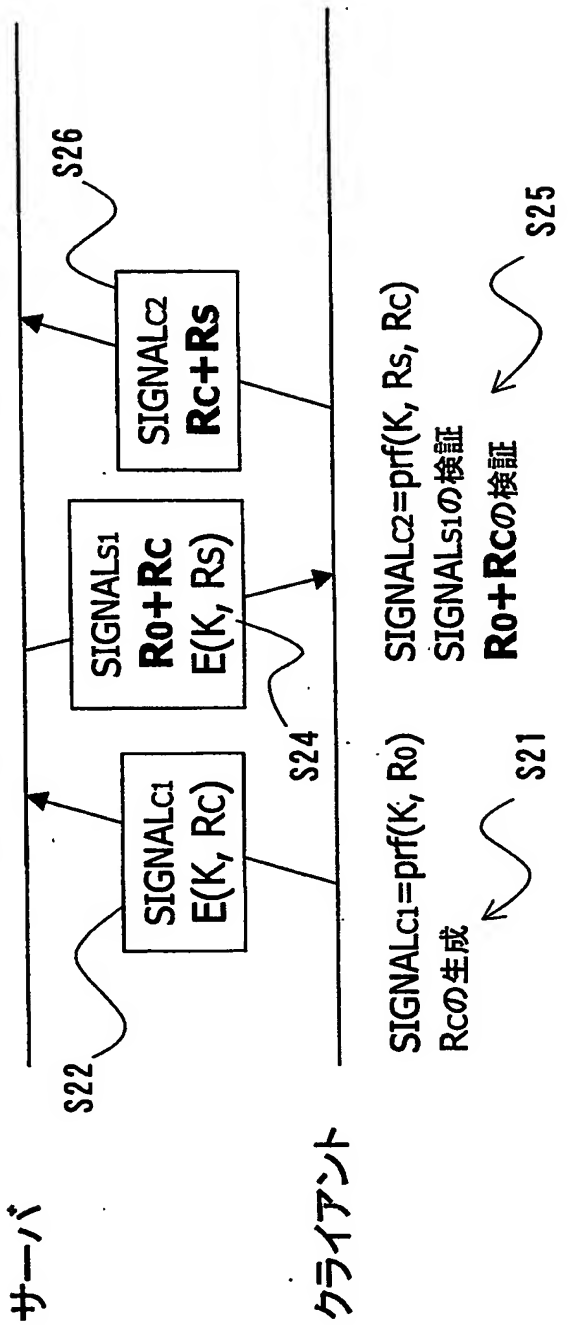
【図５】


クライアント
サーバ
P1
SIGNAL n, 1
E kn−1 (g xn , IDc, IDs, SIGNAL n, 1)
P2
クライアントの正当性を判定
P3
SIGNAL' n, 1 、 g yn
h(K n IDc, IDs, SIGNAL' n, 1)
サーバの正当性を判定
P4

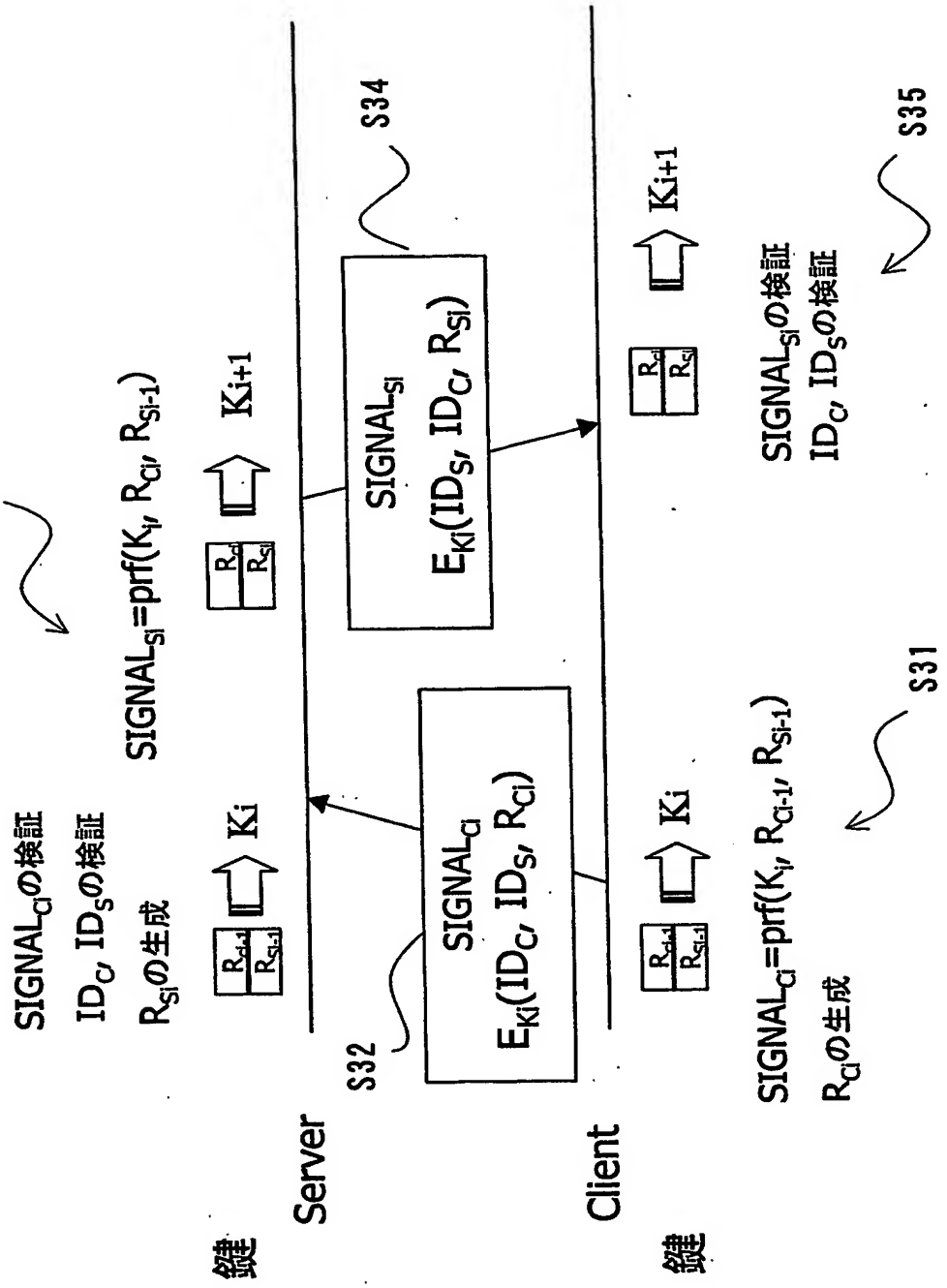
【図６】


S17
SIGNALc2の検証
Rc+Rsの検証
S16
SIGNALc2
Rc+Rs
S13
SIGNALs1=prf(K1, Rc)
Rsの生成
K2
Rs
Rc
SIGNALs1
R0+Rc
E(K1, Rs)
鍵
K1
S12
SIGNALc1
E(K1,Rc)
S14
サーバ
クライアント
K2
Rs
Rc
SIGNALc2=prf(K2, Rs, Rc)
SIGNALs1の検証
R0+Rcの検証
S15
鍵
K1
SIGNALc1=prf(K1, R0)
Rcの生成
S11

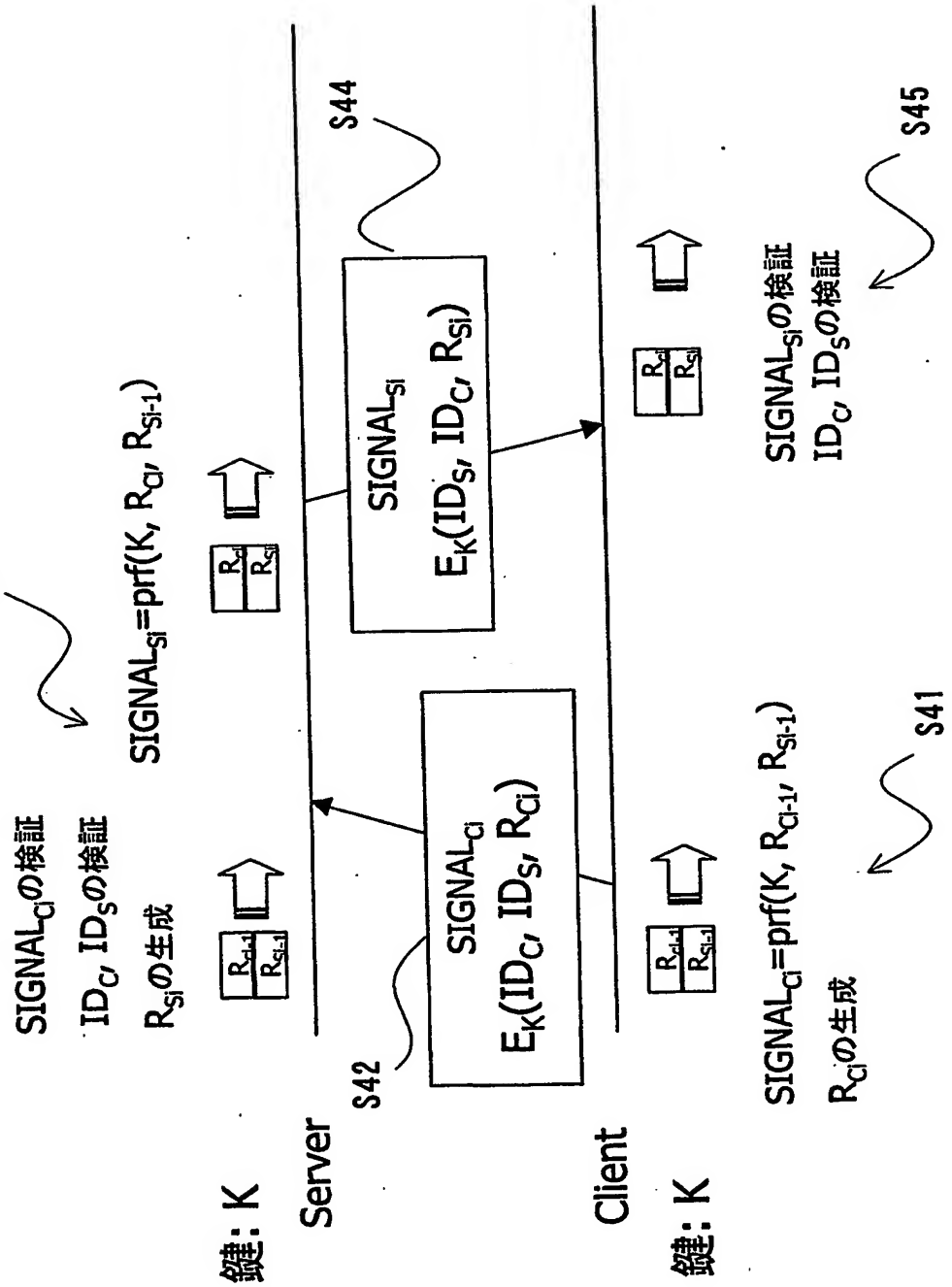
【図７】


サーバ
クライアント
S21
SIGNALC1=prf(K, R0)
RCの生成
S22
SIGNALC1
E(K, RC)
S23
SIGNALS1=prf(K, RC)
RSの生成
S24
SIGNALS1
R0+RC
E(K, RS)
S25
SIGNALC2=prf(K, RS, RC)
SIGNALS1の検証
R0+RCの検証
S26
SIGNALC2
RC+RS
S27
SIGNALC2の検証
RC+RSの検証

【図８】


S33
SIGNAL_Ci の検証
ID_C, ID_S の検証
R_Si の生成
SIGNAL_Si=prf(K_i, R_Ci, R_Si-1)
Ki
Ki+1
S34
SIGNAL_Si
E_Ki(ID_S, ID_C, R_Si)
S32
SIGNAL_Ci
E_Ki(ID_C, ID_S, R_Ci)
Server
Client
鍵
鍵
S31
SIGNAL_Ci=prf(K_i, R_Ci-1, R_Si-1)
R_Ci の生成
S35
SIGNAL_Si の検証
ID_C, ID_S の検証

【図９】


S43
SIGNAL$_{Ci}$の検証
ID$_C$, ID$_S$の検証
R$_{Si}$の生成
SIGNAL$_{Si}$=prf(K, R$_{Ci}$, R$_{Si-1}$)
S44
SIGNAL$_{Si}$
E$_K$(ID$_S$, ID$_C$, R$_{Si}$)
S45
SIGNAL$_{Si}$の検証
ID$_C$, ID$_S$の検証
S42
SIGNAL$_{Ci}$
E$_K$(ID$_C$, ID$_S$, R$_{Ci}$)
S41
SIGNAL$_{Ci}$=prf(K, R$_{Ci-1}$, R$_{Si-1}$)
R$_{Ci}$の生成
鍵：K
Server
Client
鍵：K

【図１０】

Client
Server
h, P
h, h²(P⊕n), n
S51
ID_C
S52
n
S53
c1 = h(P⊕n)⊕h²(P⊕n)
c2 = h²(P⊕(n+1))⊕h(P⊕n)
c3 = h³(P⊕(n+1))
S54
c1, c2, c3
S55
C1、C2、C3に基づいて
クライアントの正当性を検証
検証用情報および通信回数の更新

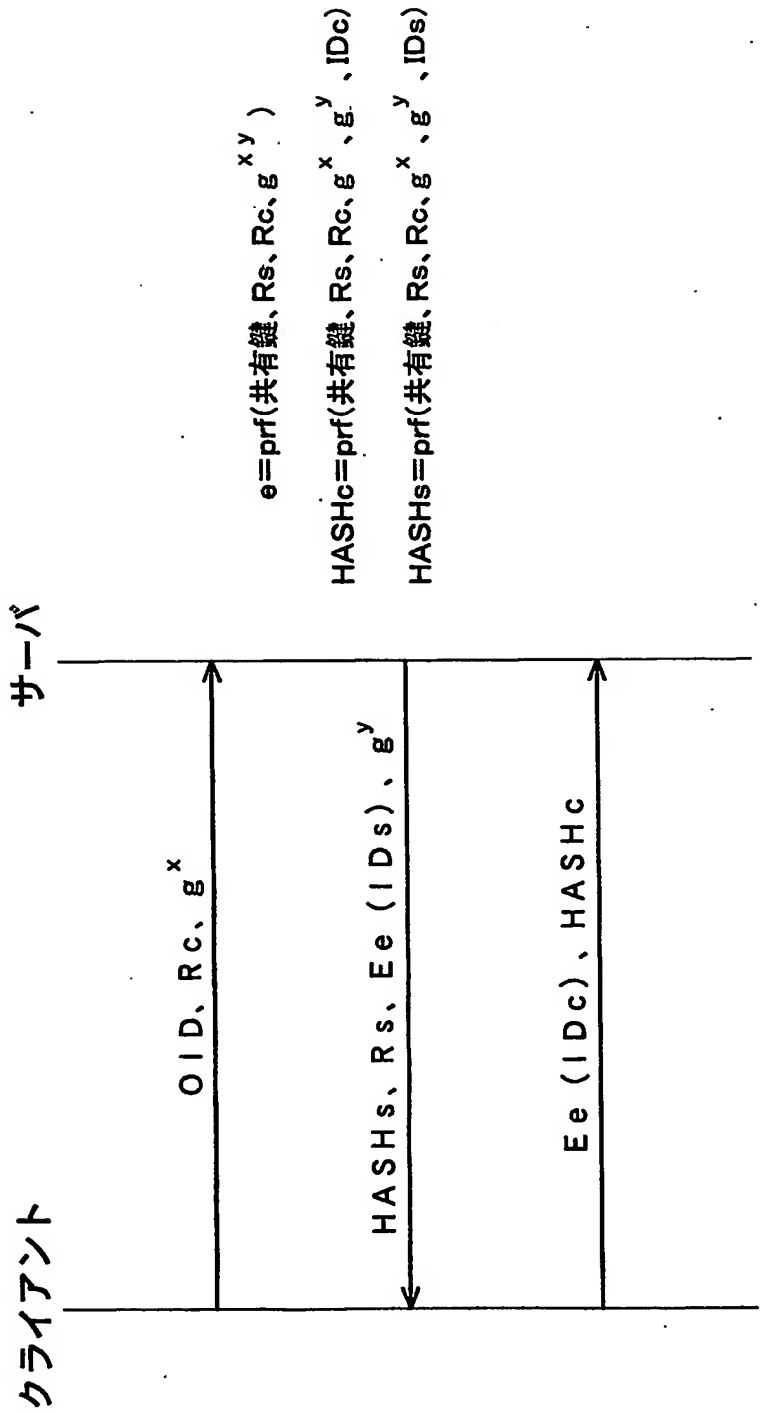
【図１１】

Client
Server
h, P
h, h²(P⊕n), n
S61
ID_C
S62
n
S63
c1 = h(P⊕n)⊕h²(P⊕n)
c2 = h²(P⊕(n+1))⊕h(P⊕n)
c3 = h³(P⊕(n+1))
SIGNAL_n = h(h²(P⊕n), n)
S64
c1, c2, c3
SIGNAL_n
S65
SIGNAL_n の検証
C1、C2、C3に基づいて
クライアントの正当性を検証
検証用情報およびセッション回数の更新

【図１２】

Client
Server
h, P, n
h, h²(P⊕n), n
S63
S64
S65
$c_1 = h(P \oplus n) \oplus h^2(P \oplus n)$
$c_2 = h^2(P \oplus (n+1)) \oplus h(P \oplus n)$
$c_3 = h^3(P \oplus (n+1))$
$SIGNAL_n = h(h^2(P \oplus n), n)$
c_1, c_2, c_3
$SIGNAL_n$
SIGNAL$_n$の検証
C1、C2、C3に基づいて
クライアントの正当性を検証
検証用情報およびセッション回数の更新

【図１３】


e=prf(共有鍵、Rs、Rc、g^xy)
HASHc=prf(共有鍵、Rs、Rc、g^x、g^y、IDc)
HASHs=prf(共有鍵、Rs、Rc、g^x、g^y、IDs)
サーバ
クライアント
OID、Rc、g^x
HASHs、Rs、Ee(IDs)、g^y
Ee(IDc)、HASHc

【書類名】　　要約書

【要約】

【課題】　盗聴が困難で安全性に優れたワンタイムＩＤの生成方法、上記ワンタイムＩＤを用いた認証方法、認証システム、サーバ、クライアントおよびプログラムを提供する。

【解決手段】　複数の装置間またはアプリケーション間の認証において一回限り使用可能な識別情報をワンタイムＩＤとして、当該ワンタイムＩＤを生成する方法である。認証を行う装置またはアプリケーションの各々において、認証が必要な所定の通信単位毎に変化する可変共有鍵を生成するとともに、この可変共有鍵を引数とする一方向関数の関数値を求め、この関数値から上記ワンタイムＩＤを生成するようにした。

【選択図】　　図４

認定・付加情報

特許出願の番号	特願２００３－０６９３７５
受付番号	５０３００４１９３９７
書類名	特許願
担当官	小野寺　光子　　１７２１
作成日	平成１５年　３月１７日

＜認定情報・付加情報＞

【提出日】	平成15年 3月14日

次頁無

出願人履歴情報

識別番号　〔８０００００３５〕

１．変更年月日　２０００年１０月１８日
［変更理由］　住所変更
住　所　福岡県福岡市東区箱崎６丁目１０番１号
氏　名　株式会社産学連携機構九州

出 願 人 履 歴 情 報

識別番号　　〔５００１９６０８７〕

１．変更年月日　　２０００年　４月２７日
［変更理由］　新規登録
住　所　神奈川県川崎市中原区新丸子９１５－１５
氏　名　株式会社エイシーエス